

Real-Time Character Animation Based on Movement Primitives from MoCap Data

Dissertation

der Fakultät für Informations- und Kognitionswissenschaften
der Eberhard-Karls-Universität Tübingen
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt von

Aee-Ni Park

aus Warendorf

Tübingen 2010

Tag der mündlichen Qualifikation: 14.07.2010
Dekan: Prof. Dr.-Ing. Oliver Kohlbacher
1. Berichterstatter: Prof. Dr. Martin A. Giese
2. Berichterstatter: Prof. Dr. Andreas Schilling

Acknowledgements

First, I would like to express my gratitude to my supervisors Professor Martin Giese and Professor Andreas Schilling for their four years of supervision and especially for their commitment to guiding me through my doctoral research, as well as the many helpful comments and suggestions regarding the expository and critical aspects of my thesis.

I am grateful to Professor Jean-Jacques Slotine for being a valuable source of ideas and motivation. Regarding to this, I thank also my colleagues: Albert Mukovskiy, Enricho Chiovetto, Andrea Christensen, Dominik Endres, Falk Fleischer, Winfried Ilg, Lars Omlor, Karsten Rohweder, Susanne Smidt, Nick Taubert, and Florentin Vintila.

I am also greatly indebted to all my friends, in particular to my best friend Nora Kubani, who supported me in any respect during the completion of the project. Especially, I owe my deepest thanks to David Voss, for unconditional support during the last years. My heartily thanks to Pedro Perera who brought me laugh even in the most difficult time of my life.

Last but not least, my love and gratitude to my wonderful parents for their dedication and their support during my whole life. Thank you for believing in me.

Zusammenfassung

Die Erzeugung realistischer und komplexer Körperbewegungen in Echtzeit ist für Computergraphiker und Robotiker eine schwierige und anspruchsvolle Aufgabe. Eine Animation realistischer Bewegungsabläufe setzt voraus, dass die zugrunde liegenden Trajektorien vor dem Hintergrund einer Vielzahl gegebener Freiheitsgrade möglichst detailliert und genau modelliert werden. Gleichzeitig müssen echtzeitfähige Animationen flexible Systeme und Mechanismen enthalten, um reaktiv und dynamisch mit der Umwelt zu kommunizieren bzw. sich ihr anzupassen. Solche Online-Systeme eignen sich für die Selbstorganisation dynamischer, komplexer Szenen, bei denen mehrere virtuelle Charaktere autonom miteinander interagieren.

Diese Arbeit stellt einen neuartigen Ansatz zur Online-Synthese realistischer menschlicher Körperbewegungen vor, der durch Konzepte aus dem Forschungsfeld der Motor-Control inspiriert wurde. Anhand von unüberwachten Lernmethoden können ganzkörperliche Bewegungen durch die Überlagerung von Bewegungsprimitiven –sogenannten Synergien– approximiert werden, welche auf Daten des Motion Capture Verfahrens beruhen, wobei ein neuartiger blind source separation Algorithmus zur Anwendung kommt. Im Vergleich zu den Standardverfahren der PCA und ICA stellt dieser Ansatz hinsichtlich der Dimensionsreduktion von Daten eine signifikant kompaktere Repräsentation dar.

Werden überwachte Lernmethoden benutzt, um ein echtzeitfähiges Modell zu erhalten, so können die Quellensignale als stabile Lösungen niedrigdimensionaler, nichtlinearer dynamischer Systeme dargestellt werden. Auf diese Weise können periodische und nichtperiodische Bewegungsabläufe generisch erzeugt werden, wobei ein hoher Realismusgrad mit einer nur sehr geringen Anzahl von Synergi-

en gewährleistet wird.

Die Anwendung eines neuen Verfahrens der Stabilitätsanalyse, aus der Contraction Theory ermöglicht das Erstellen komplexer Netzwerke von dynamischen Primitiven mit einer stabilen Systemarchitektur. Während die Erzeugung von Stabilitätseigenschaften dynamischer Systeme im Bereich der Control Theory und Robotik zu den Kernthemen gehören, wird Stabilität im Kontext der Computeraanimation kaum berücksichtigt. Grund dafür ist u.a. die Architektur dynamischer Systeme, die für realistische Modellierung von menschlichen Körperbewegungen äusserst komplex ist, v.a., wenn interaktives Verhalten zwischen multiplen virtuellen Charakteren dargestellt werden soll.

Vor diesem Hintergrund wird der vorgestellte Ansatz durch die Einbeziehung von Methoden aus der Contraction Theory erweitert, um koordinierte menschliche Bewegungen online zu simulieren. Auf diese Weise können Stabilitätsproblemen bei der Charakteranimation systematisch betrachtet und gelöst werden. Daraus resultierend können globale Stabilitätseigenschaften gewonnen werden, auch bei Systemen, die viele Charaktere oder nichtlineare Interaktionsmodule enthalten.

Zusammengefasst ergibt sich:

- Bewegungsprimitive bzw. Synergien werden aus Motion Capture Daten von ganzkörperlichen Bewegungen erzeugt.
- Die erhaltenen Trajektorienmodelle werden in ein echtzeitfähiges System transformiert.
- Mit diesem System können äusserst realistische Animationen menschlicher Bewegungsabläufe generiert werden.
- Interaktives und koordiniertes Verhalten unter virtuellen Charakteren kann simuliert werden.
- Um die Stabilität der dynamischen Architektur zu gewährleisten, werden Methoden aus der Contraction Theory angewandt.

Abstract

The synthesis of realistic and complex body movements in real-time is a challenging task in computer graphics and in robotics. High realism requires accurate modeling of the details of the trajectories for a large number of degrees of freedom. At the same time, real-time animation necessitates flexible systems that can react in an online fashion and, therefore, adapting to external constraints. Such online systems are suitable for the self-organization of complex behaviors due to the dynamic interaction among multiple autonomous characters in the scene.

A novel approach for the online synthesis of realistic human body movements is hereby presented. The proposed model is inspired by concepts from motor control as it approximates full-body movements by the superposition of lower-dimensional movement primitives -synergies- that are learned from motion capture data. For this purpose, a blind source separation algorithm is applied and provides significantly more compact representations than standard approaches for dimension reduction, such as PCA or ICA.

When applying supervised learning methods, these source signals are further associated or ‘replaced’ with the stable solutions of low-dimensional nonlinear dynamical systems (dynamic primitives) in order to obtain a real-time architecture. In this way, the learned generative model can synthesize periodic and non-periodic movements, achieving high degrees of realism with a very small number of synergies.

The application of a new type of stability analysis (contraction theory) permits the design of complex networks of such dynamic primitives, resulting in a stable overall system architecture. The design of stability properties of dynamical systems

has been a core topic in control theory and robotics, but has rarely been addressed in the context of computer animation. One potential reason is the enormous complexity of the dynamical systems that are required for the accurate modeling of human body movements, and even more for the interaction between multiple interacting agents.

In this context, the approach to the online simulation of realistic coordinated human movements is extended by introducing contraction theory as a novel framework that permits a systematic treatment of stability problems for systems in character animation. It yields tractable global stability conditions, even for systems that consist of many nonlinear interacting modules or characters. To summarize:

- Movement primitives or synergies from full-body motion capture data are learned.
- The transformation of such trajectory models into a real-time capable system is obtained.
- With this process, one is able to generate high-quality animation of complex human movements.
- The simulation of interactive behavior and coordinated crowd animation is enabled.
- To ensure the stability of the dynamical architecture, contraction theory is applied to compute stability properties.

Contents

1	Introduction	2
1.1	Background	3
1.2	Main Contribution	7
1.3	Context and Related Work	8
1.3.1	Motion Synthesis	8
1.3.1.1	Kinematical Models based on Motion Capture	8
1.3.1.2	Physical Models	10
1.3.1.3	Hybrid Models for Interactive Character Animation	11
1.3.1.4	Dimensionality Reduction in Character Animation	12
1.3.2	The Idea of Synergies	13
1.3.2.1	Synergies	13
1.3.2.2	Humanoid Robots	15
1.3.3	Dynamics for Collective Behavior	15
2	Nonlinear Dynamical Systems	18
2.1	Synchronization	19
2.1.1	Self-Sustained Oscillator	20
2.2	Limit Cycle Oscillators	22
2.2.1	Phase Space	22
2.2.2	Limit Cycle	22
2.2.3	Van der Pol Oscillator	23
2.2.4	Andronov-Hopf Oscillator	24
2.3	Contraction Theory and Partial Contraction Theory	25

2.3.1	Contraction Theory	26
2.3.2	Contraction of Dynamical Systems	27
2.3.3	Partial Contraction	29
2.4	Contraction Analysis for Coupled Nonlinear Dynamical Systems	32
2.4.1	Andronov-Hopf Oscillator	34
2.4.1.1	Symmetric Coupling of Two Oscillators	35
2.4.1.2	Symmetric Coupling of Three Oscillators	38
2.4.1.3	Symmetric All-to-All Coupling of N Oscillators	38
2.4.2	Symmetric Couplings: More General Structure	40
2.4.3	Leader-Group Coupling	42
2.5	Distance-Frequency Control	46
2.5.1	Analysis	47
2.5.1.1	Linear Approximation of Kinematics	47
2.5.1.2	Nonlinear Approximation of Kinematics	47
3	Blind Source Separation	50
3.1	General Approach to Blind Source Separation	51
3.1.1	BSS For Motion Analysis	52
3.2	Instantaneous Blind Source Separation	54
3.2.1	Principal Components Analysis	54
3.2.2	Independent Component Analysis	56
3.2.3	Non-Negative Matrix Factorization	57
3.3	Anechoic Demixing using Wigner-Ville Transform	58
4	Real-Time Architecture for Character Animation	60
4.1	Data Acquisition	60
4.1.1	Motion Capture System	60
4.1.2	Joint Representations of the Avatars	62
4.1.2.1	Joint angle computation	63
4.1.2.2	Axis Angle	65
4.2	Data Reduction Based on BSS	66
4.2.1	Database of Recorded Data	66
4.2.2	Learning Movement Primitives	67
4.2.2.1	Approximation Quality	68
4.3	Dynamical System for Trajectory Generation in Real-Time	69
4.3.1	Attractor Dynamics	71
4.3.2	Dynamic Primitives	71
4.3.3	Mapping between Attractor Solutions and Source Signals	73
4.3.4	Stabilization by Dynamic Coupling	77

5	Dynamic Character Animation	79
5.1	Autonomous Agents for Interactive Character Animation	80
5.1.1	Dynamic Style Morphing	81
5.1.1.1	Integration of Periodic and Non-Periodic Synergies	84
5.1.1.2	Modulation of Walking Speed	87
5.1.2	Character Propagation	88
5.2	Crowd Animation	90
5.2.1	Navigation	90
5.2.1.1	Navigation Algorithm Outline	92
5.2.1.2	Control of Walking Direction	95
5.2.2	Self-Organized Behavior in Crowds	98
5.2.2.1	Coordinated Crowd Behavior	99
6	Stability Properties in Character Animation	106
6.1	Collective Behavior Fulfilling And Violating Contraction Bounds .	108
6.1.1	All-to-All Symmetric Coupling	109
6.1.2	Chain Coupling	110
6.1.3	Leader-Group Interaction	111
6.1.4	Theoretical vs. Empirical Convergence Rates	114
6.1.5	Distance Frequency Coupling	116
7	Testing Environment	118
7.1	Visualization	118
8	Conclusion and Outlook	123
	Appendix	128
.1	Notation	128
.2	Tables	130
.3	Authored Publications	132

List of Figures

1.1	Performer’s motion drove the movement of Gollum’s body through motion capture [HbT08].	3
1.2	The Uncanny Valley: ‘For the animators of films and video games, creating a truly human looking and acting character has long been the holy grail.’ But making characters close-to-real and yet not-real-enough leaves them in what is called the ‘uncanny valley’ where audiences find those characters unsettling, unnatural and zombie-like’ [Med10].	4
1.3	Crowd animation. [TCP06]	11
1.4	A character switches automatically to obstacle avoidance mode and navigates around the obstacles [TLP07].	12
1.5	Shared synergies are extracted from the entire dataset whereas the other behavior-specific synergies are extracted from only the muscle patterns from two behaviors (jump-swim and jump-walk) or a single behavior (swim and walk)[dB05].	14
1.6	Flocking example of a fish school: ‘Tornado unter Wasser’. [HbF09]	16
2.1	Two pendulum clocks coupled through a beam (Orginal picture from Huygen) [Huy67, PRK03].	19
2.2	Two non-identical pedulum clocks: a) The pendula are in different positions $\alpha_1 \neq \alpha_2$ at some arbitrary moment of time. b) They differ in their periods $T_2 > T_1$	20
2.3	a) Synchronization in phase. b) Synchronization in anti-phase. . .	21
2.4	Limit-cycle oscillator	23
2.5	Van der Pol oscillator viewed in the time domain (left) and in the phase space (right).	24

2.6	The radial dynamics has three stationary solutions where $\dot{r} = f(r) = 0$. Whereas the fixed point at $r_1 = 0$ is unstable, the other two $r_2 = \pm 1$ are stable.	25
2.7	Two trajectories of a dynamical system and the virtual displacement.	28
2.8	Flow-invariant <i>linear subspace</i> \mathcal{M} , such that $\forall t : f(\mathcal{M}, t) \subset \mathcal{M}$. Any trajectory starting in \mathcal{M} remains in \mathcal{M}	31
2.9	Diffusive coupling with equal coupling gains K , where \mathcal{N}_i define a set of neighbors of i	32
2.10	a) Two uncoupled oscillators show no synchronization behavior; b) Two bidirectionally coupled oscillators synchronize.	37
2.11	All-to-all coupling of N oscillators, with equal symmetrical coupling gains k	39
2.12	Symmetric coupling. a) Chain and b) ring coupling of N oscillators.	41
2.13	Star coupling	42
2.14	Leader-group coupling: Single oscillator feeds unidirectionally into all other N oscillators with the same coupling strength k	43
2.15	a) Two oscillators with opposite initial conditions $(1, -1)$ are uncoupled resulting in an anti-synchronization behavior. b) Two unidirectionally coupled oscillators synchronize.	45
2.16	Following the Leader: $z_1(t)$ and z_2 define the position of the follower and the leader at time t	46
2.17	Propagation velocity dependent on the gait cycle phase ϕ	48
3.1	The <i>cocktail-party problem</i> presents the classical BSS problem since the number of speakers is larger than the observed mixtures recorded by the two microphones x_1 and x_2 . Hence, this example represents an under-determined problem because several simultaneously active signal sources can be separated at different spatial locations by assuming mutual independence of the sources.	51
3.2	Extraction of primitives by using ICA on individual joint angles. Source signals of individual joints have a similar shape, but are time-shifted against each other.	53
3.3	Generative mixing model: Superposition of independent shift-invariant source signals.	54
3.4	PCA projects the data along the directions where the data varies the most. These directions are determined by the eigenvectors of the covariance matrix corresponding to the largest eigenvalues. The magnitude of the eigenvalues corresponds to the variance of the data along the eigenvector directions.	55

4.1	From a real performer to a virtual character. Left to right: Capturing human walking; 3D positions of recorded data; joint angle computation and mapping them onto a simple character; and animate a virtual human.	61
4.2	Motion capture environment. Vicon cameras record the performer's motion.	62
4.3	According to the Plug-In Gait marker set, gray spheres show the placement of the markers. Red dots indicate the location of virtual markers that can be computed using additional anatomical measurements.	63
4.4	Representation of the hierarchical structure of our model.	64
4.5	Four source signals were extracted from the joint angles trajectories of the motion capture dataset. Sources 1 – 3 represent the periodic and the 4th source the non-periodic movements.	68
4.6	Comparison of different blind source separation algorithms of periodic and non-periodic movements. The approximation quality Q is shown as a function of the number of sources for traditional blind source separation algorithm (PCA) and the applied new algorithm.	69
4.7	How can the extracted source signals be encoded for a real-time application?	70
4.8	Online implementation of delays. a) Direct implementation introduces explicit delay lines, resulting in a complex system dynamics that is difficult to control. b) Approximation by a rotation in the phase space, defined by the instantaneous orthogonal transformation $\mathbf{H}_{\tau_{ij}}$ in the phase plane of the oscillator, avoids a dynamics with delays.	73
4.9	Illustration of the dynamic architecture for real-time animation. Periodic and non-periodic movements are generated by dynamic primitives. The solutions of these dynamical systems are mapped onto the source signals by a nonlinear mapping that models the time delays and a nonlinear transformation that is learned by SVR. Joint angle trajectories can be synthesized by combining the signals linearly according to the learned mixture model 4.1. A kinematic model converts the joint angles into 3-D positions for animation.	74
4.10	Demo simulates the comparison between approximated and original motion. Furthermore, the comparison between different emotions can be seen in the following movies: MovieAngry , MovieSad , MovieHappy	76

4.11	Every avatar is driven by three limit cycle oscillators. The coupling of the oscillators ensure coordinated behavior and prevent the influence of external perturbations by dynamical noise.	77
4.12	Comparison between two characters which are consisting of <i>uncoupled</i> and <i>coupled</i> dynamic primitives. a) Dynamic noise is included, in order to simulate e.g. disturbances or environmental noise. b), c) While the movement of the left character becomes uncoordinated, the coupled character on the other hand gets stabilized. The corresponding movie can be found under DEMO	78
5.1	Virtual humans present a standard element in most modern entertainment software.	79
5.2	This figure present examples of the relationships between autonomous and interactive character animations	80
5.3	Depending on specific weight components, different motion styles can be simulated. Example: Boy walking sadly on the left side and happy on the right side.	81
5.4	Morphing between neutral and happy walk.	83
5.5	Movement primitives can be combined with specific weight components for individual styles. By morphing between them, a huge repertoire of different movement styles is given.	84
5.6	Simulation of an arm up movement by recombining the learned synergies with dynamically changing morphing weights. Cyan line: Periodic movements of the feet. Blue line: Fourth non-periodic source. Green line: External trigger signal	85
5.7	Dancing figures from a folk dance. The sequence was generated online by blending and recombination of the learned synergies with dynamically changing morphing weights in matlab a) as well as with an animation software b). The corresponding movie can be found under Movie (See text for further details.)	86
5.8	Following behavior realized by two coupled avatars. a) With the distance exceeding a certain threshold the smaller avatar accelerates to catch up with the leader b) Eigenfrequency ω_0 dependent on the distance $d(t)$ between different characters. [cf. Movie]	87
5.9	Pelvis scheme: In every time-step we compute the differential changes in the pelvis xy -direction angle in respect to the foot-on-the-ground xy -direction angle.	88
5.10	Humans constantly adjust their paths to navigate through a complex environment.	91

5.11	Navigation dynamics depending on: a) Goal-finding term, b) instantaneous obstacle-avoidance term, and c) predictive obstacle-avoidance term.	93
5.12	Avoidance behavior and change of emotional style. a) Avatars starting from different positions with a sad emotion are heading towards their goals (red circles). b) At the meeting point the emotional styles change to happy. In addition, the characters avoid each other. DEMO	95
5.13	Simulation of stationary and dynamic obstacle avoidance, while changing the emotional styles and speed interactively. a) The avatars' goal areas are positioned on the opposite side of the scene's floor. b), c), d) Static obstacles (columns) and dynamic obstacles (other avatars) have to be avoided simultaneously. e), f) After avoiding all obstacles, the characters proceed heading towards their goal areas. The demo, corresponding to the described scene can be seen in DEMO	96
5.14	Avoidance behavior and change of emotional style. Three avatars, starting from the left side, change their emotion from happy to sad while proceeding to their goals. A second group of avatars starting from the right side change their emotions from sad to happy while avoiding the opposing group. DEMO	97
5.15	Coupling of multiple avatars, each of them comprising three coupled oscillators (Osc1, Osc2, Osc3), permits the simulation of the behavior of coordinated crowds.	99
5.16	Autonomous synchronization of gait patterns within a crowd. (a) Avatars start with self-paced walking and are out of phase. After a transitory period (b), the gaits of the characters become completely synchronized (c).	100
5.17	This example video shows the comparison of a synchronization behavior of a model, which is generated by our method with a standard approach such as PCA. See text for more details. [DEMO]	101
5.18	Phase plot of the first (principal) oscillators of two avatars that are coupled to a leader. The diagram shows the state space of the oscillator in the $[y(t), \dot{y}(t)]$ plane. The green line corresponds to the avatar driven by five PCA components, and the red line to the avatar animated with three sources extracted with our algorithm. The coupling is switched on after a short starting period.	102

5.19	Simulation of a 'folk dance' behavior is organized within four zones: 1) Avatars within the corridor move in synchrony, simulated by coupling the corresponding oscillators. 2) Leaving the corridor, the avatars of the individual dancing couple become separated and start to move along curved paths. Their emotional style changes from happy to neutral. 3) Avatars walk asynchronously and 'hurry up' to reach the entrance of the corridor, simulated by increasing the eigenfrequency of the oscillators. 4) When the avatars approach the entrance of the corridor their walking speed decreases, and they need to get in synchrony again with the appropriate leg. This can be simulated by appropriate adjustment of the oscillator frequencies. [DEMO]	103
5.20	Self-organized formation behavior of interacting characters ('soldiers') in crowds. Different coupling methods and navigation dynamics have been applied to observe the coordinated behavior. [DEMO1] or [DEMO2]	105
6.1	Illustration of a large crowd animation in video games (Napoleon: Total War).	106
6.2	Group of characters that are all-to-all coupled and fulfill the contracting conditions ($k = 0.334$). The characters start with random initial condition (a) and adapt to the other step phases (b, c) and until they are synchronized (d, e). [Movie_1]	109
6.3	Group of characters that are all-to-all but with a violating contracting condition ($k=0.111$). The characters start with different step cycles (a) and do not synchronize (b, c). [Movie_2]	110
6.4	Group of characters that are all-to-all but with a violating contracting condition ($k=-2.0$). Due to the coupling gain, no coordinated behavior can be achieved and in addition, the walking movement becomes even unnatural (a-c). [Movie_3]	110
6.5	Group of characters with a chain structure, while fulfilling the contracting condition ($k = 1.0$): Starting with random initial conditions (a), the phase steps are adapted from each other, (b) which result in a coordinated group behavior (c). ([Movie_4])	111
6.6	Group of characters with a chain structure but violating the contracting condition: Characters start with individual step-cycles (a) and do not synchronize (b, c). The corresponding demo can be found here: [Movie_5]	111

6.7	Leader-group interaction: The contraction condition is not fulfilled ($k = 0.01$, $\alpha = 0$), showing no coordinated behavior. The characters start with uncoordinated, individual step-phases (a) and do not synchronize (b, c). [Movie_6]	112
6.8	Leader-group interaction with a fulfilled contraction condition. A synchronization behavior can be observed. [Movie_7]	112
6.9	Leader-group interaction: a) At the beginning, the characters start with the initial condition (yellow bar). After introducing couplings the group becomes coordinated (green bar). The leader experiences a perturbation by changing its frequency (red bar) and the follower adapts to the leaders phase again. b) Instead of the leader, a member of the group gets perturbed in an identical way and relaxes back to reestablish synchrony with the other members of the group. [Movie_8] and [Movie_9]	113
6.10	a) Dispersion of the phase of the oscillators, averaged over 100 simulations with random initial conditions, as function of time (gait cycles). After an offset time, during which the dispersion remains relatively constant, it decays exponentially. Convergence rates were estimated by fitting linear functions to this decay. b) Offset times (in gait cycles) as function of the coupling strength. (End of offset time interval was defined by the point where the regression line crosses the level $\hat{R} = 1$.)	115
6.11	a) The relationship between convergence rate and coupling strength k for different types of coupling graphs. b) Slopes of this relationship as function of the number N of Hopf oscillators, comparing simulation results (indicated by asterisks) and derived from the theoretical bounds (Section 2.4.1).	115
6.12	Distance Frequency Coupling: Follower $F1$ was coupled with positive and follower $F2$ with a negative coupling force to the leader L , whose frequency is dependent on the distance. $F1$ converges to the leader by increasing the speed, where the behavior of $F2$ diverges to minus infinity, which shows a back walk. [Movie_10]	116
7.1	Modeling avatar: Geometrical shapes were attached to the computed marker positions.	119
7.2	Motion trajectories have to be mapped onto a Biped 3D-model (left), which defines the animation of a character (right).	120
7.3	This screenshot shows an animation of a dance scenario in 3D-Studio Max. Characters contain Biped models (bone skeleton), which comprise the motion information	121

List of Tables

- 1 Marker names and definitions of the Vicon Plug-In Gait model . . 130
- 2 Skeleton segments, and attached coordinate frames [x, y, z] 131

CHAPTER 1

Introduction

Body motions, in particular human full-body movements, are complex procedures which require e.g. the coordination of the skeleton, the muscles and the control of multi-joint movements. Nevertheless, since the generation of realistic human motion is highly expected in various areas such as computer graphics, robotics, or biomechanics, scientists are faced with the challenge of motion creation in order to satisfy the community.

The controllability of real-time character animation that approaches the quality of natural-looking movement based on motion capture systems is still an unsolved problem for a number of reasons.

A key reason is its high dimensionality: Characters have a relatively high number of degrees of freedom and are often provided as a set of different motion styles, making the search for the appropriate control parameters hard in order to obtain a dynamic animation. If, in addition to that, interactive control algorithms (e.g. navigation algorithm) for generating reactive behavior in the dynamic environment were included, the control architecture would become even more complex. The movements are often simplified at the expense of quality with the purpose of preventing infeasibility as well as saving computational cost.

In this context, we try to address these difficulties and show an efficient method for the representation of human movements based on learned primitives from motion capture data, which results in a simple dynamical control architecture for generating human motion.

The novel method is suitable for the real-time simulation of natural looking human movements combined with the dynamic variation of different movement styles. For an autonomous and self-organized behavior in e.g. crowd animation, we further included modules like navigation algorithms, which are crucial for interactive manners in real-time animation. As to ensure the stability of e.g. autonomous crowd scenarios, the stability constraints were analyzed by using the concept of contraction theory.

1.1 Background



Figure 1.1: Performer's motion drove the movement of Gollum's body through motion capture [HbT08].

Human animation in computer graphics with important applications as in humanoid robots, and especially in the film and game industry, have become increasingly popular. In famous Hollywood blockbusters such as *Titanic*, *Star Wars* and *Lord of the Rings* (Fig.1.1), many synthetic scenes involve virtual humans like in special effects. With character animation, for instance, dangerous action scenes can be simplified and crowd simulations in films can be animated without organizing hundreds of actors.

The interest in the area of human animation never decreased. On the contrary, since the observer itself is experienced with human movement and will notice immediately any artifacts in motion, the task of simulating natural movements becomes an important research field. In addition, human beings represent in general a very critical and skeptic audience when they are confronted with human motion in humanoid robots or animated characters. Figure 1.2 describes the emotional reaction to humanoid robots or animated characters, who look and act like actual humans: At first, a humanlike appearance in motion leads to a positive emotional reaction. At a certain degree of realism the character become 'eerily' lifelike, and the reaction becomes strong revulsive. This is known as the 'uncanny valley phenomenon' [Mas70]. Due to this fact, the simulation of 'realistic' looking human animations in order to satisfy the audience, became even more challenging and sophisticated. However, only few people are aware of the non-

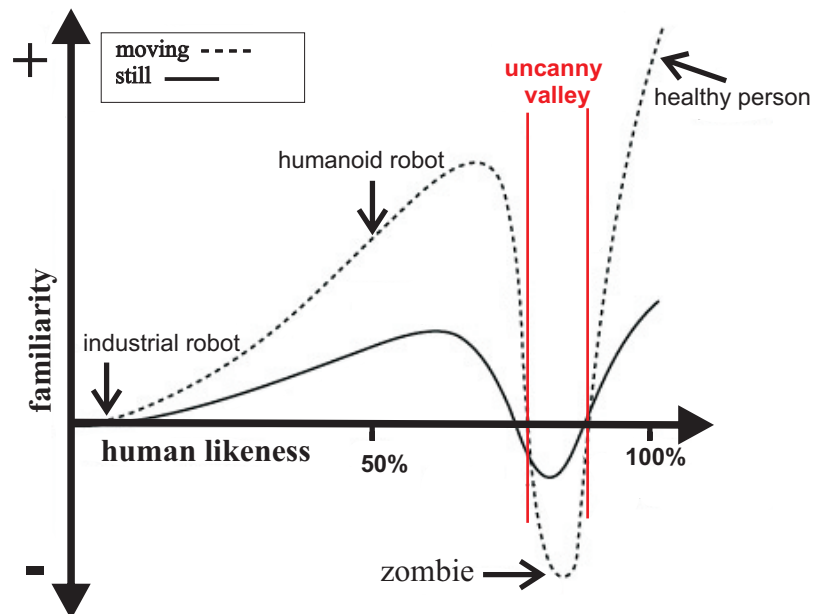


Figure 1.2: The Uncanny Valley: ‘For the animators of films and video games, creating a truly human looking and acting character has long been the holy grail.’ But making characters close-to-real and yet not-real-enough leaves them in what is called the ‘uncanny valley’ where audiences find those characters unsettling, unnatural and zombie-like’ [Med10].

trivial process for obtaining complex behavior in character animation, where the realism requires the accurate modeling of the detailed trajectories of the joints with different techniques based on i.e. kinematics, dynamics and biomechanics. Considering that every joint has its own number of degrees of freedom, the con-

Control of complex multi-joint movements is typically highly redundant, i.e. the same behavior for the end effector like the hand can be accomplished with many different combinations of joint angles, or muscle commands. This central challenge of controlling multi-joint movements in the computer animation community addresses a well-known problem and is the reason why animators find 3-D character animation so tedious. The two main approaches for character animation in the computer graphics are based on *kinematic* and *physical* modeling.

In this manner, the first approach, **kinematic animation**, is based on motion capture data. Motion capture has become the standard approach for modeling naturalistic movement, since appropriate movements can be recorded offline and then retargeted to the relevant kinematic model. In addition, editing or blending methods can be applied onto the recorded movements in order to modify the movement style, or to make the synthesized movements compatible with constraints from the animation, which can be achieved for example through setting kinematic boundary conditions. This approach results in highly realistic animations, but requires tedious post-processing of the recorded motion capture data and makes this sort of process not real-time capable since the data from optical systems is often error-prone and noisy. Although morphing techniques like key-frame animation are applied, the concatenation of captured motion segments for creating a longer scene has been often corrected by hand.

The second approach, **physics based animation**, is mostly used for real-time applications, such as in computer games or in robotics, where the characters or robots have to interact with their dynamic environment immediately and permit an online synthesis of character behavior. Given that the simulation of scenes has to take the many interactive and autonomous agents in crowds into consideration, the underlying character models are often very simplified and lacking subtle details of realistic human body movement to achieve manageable complexity of the system dynamics and dynamics simulations.

In the more current applications, especially in video games, there is a rising trend towards **combining kinematical and physical** simulations. In this way, the simulation of realistic human behavior using kinematic models combined with physics based modeling is realized, nevertheless they are still associated with high computational cost. However, the existing approaches are characterized by high-dimensional and complex underlying dynamic control architectures, whose design requires substantial expertise from the animator.

In this case, our approach is based on a simple dynamical architecture for generating complex human movements that integrate the information learned from motion capture data, and in which we avoid a detailed simulation of the human

body dynamics. This process, however, results in real-time animation with a high degree of realism.

To accomplish this goal, our approach was inspired by motor control in biological systems: It has been a classical assumption that complex motor behavior is generated by appropriate combination of simpler movement primitives or *synergies*. Synergies specify lower-dimensional control units that typically encompass only a subset of the available degrees of freedom. Decomposition in such low-dimensional sub-units has been proposed as an alternative to solve the 'degrees-of-freedom problem', which arises in the synthesis and control of movements in effector systems with many degrees of freedom.

We extend this idea to the computer graphics by learning these movement primitives from motion capture data, and describing them using structurally stable but simple dynamical systems. Consistent with related approaches in robotics and biology, this method generates complex movements by combining the learned movement primitives. Coordinated behavior of multiple characters can then be generated by self-organization. The resulting system architecture is rather simple, despite needing a suitable treatment of dynamical properties to obtain a stable behavior. While the design of stability properties is a central topic in robotics, it is rarely addressed in character animation, even though, the application of dynamical systems in computer animation is a significant domain for the simulation of autonomous and collective behavior of many characters, i.e in crowd animation.

Some work, primarily in robotics, has been inspired by observations in biology: Coordinated behavior of large groups of agents such as flocks of birds, results from the dynamic interactions among individual agents without requiring a central mechanism that ensures coordination. It is well-known that such behaviors can be analyzed efficiently within the framework of nonlinear dynamics. This makes it interesting to exploit the underlying principles for the automatic synthesis of collective behavior in computer animation.

So far, the design of such technical systems has been often heuristic, exploiting empirical results from simulations instead of deriving the system parameters from theoretical results of the system dynamics. However, the controlled engineering of such system makes more systematic theoretically founded approaches highly desirable. A critical limiting factor in this context is the complexity of the dynamical models for individual agents or characters, which often makes a systematic treatment of stability properties infeasible.

In this context, we additionally introduce *contraction theory* as a novel framework, which permits a systematic treatment of stability problems for systems in character animation. It yields tractable global stability conditions, even for sys-

tems that consist of many nonlinear interacting modules or characters. We obtain system dynamics that can be analyzed, even for situations with multiple characters, when exploiting models that are based on the learned movement primitives.

1.2 Main Contribution

The contribution of this thesis approach different aspects of character animation in real-time and can be combined for animating interactive crowd animation. After a brief discussion of related approaches, the thesis is structured in the following chapters.

Nonlinear Dynamical Systems (Chapter 2)

We first give a general introduction of self-sustained oscillators as an example for nonlinear dynamical systems and describe their tendency to synchronize. The synchronization behavior can be seen as a result of coupling individual oscillators in an appropriate way. In order to achieve stabilization criteria for an overall stable system, we discuss the basic concepts of contraction theory, which provide analysis tools to investigate the stability of complex nonlinear systems. Furthermore, stability properties of systems with different kinds of couplings can be computed for the design of stable interactive character animations as partial contraction analysis is applied.

Dimension Reduction Techniques (Chapter 3)

The simulation of complex human motion for character animation is often tedious and deals with high-dimensional data, which can be simplified by reducing the dimensionality into lower dimensional subspaces. In this chapter, we give a short overview of blind source separation algorithms for instantaneous and convolutive mixtures. The latter is used to obtain a more compact motion representation and models our motion trajectories by mixtures of typically very few hidden source terms with time-delays (anechoic mixtures).

Real-Time Character Animation (Chapter 4)

Inspired by the concept of 'synergies', we approximate movement primitives from full-body motion capture trajectories, comprising subsets of degrees of freedom. Next, a real-time-capable system for character animation can be realized by introducing a mapping between the stable solutions of limit-cycle oscillators and the extracted source signals. Appropriate couplings among the dynamical systems are introduced and ensure the coordination of the degrees of freedom within individual characters.

Dynamic Character Animation (Chapter 5)

Methods are invented to obtain interactive character animation; Different motion

styles can be animated by morphing between different emotions and movement styles using linear interpolation. We also introduce appropriate couplings among avatars, which allow the realization of interactive behavior, such as following or group synchronization. Concerning the dynamic environment: To be able to avoid static and moving obstacles, navigation algorithms have to be implemented. The blending between examples of straight and curved walking is used for simulating the walking corresponding to the heading direction of the character.

Stability Properties: (Chapter 6)

When exploiting models that are based on learned movement primitives, we obtain a system dynamics that can be analyzed, including situations with multiple characters. Considering contracting boundaries it is now possible to obtain different kinds of desired self-organized scenarios by choosing the corresponding coupling methods and coupling strengths. Moreover, our system is tolerant towards increasing the amount of characters, thus, granting an efficient handling of the computational resources. We demonstrate how dynamic couplings can be introduced that stabilize the coordination within single characters, and which are suitable for the simulation of coordinated behavior of multiple avatars.

1.3 Context and Related Work

In the following section, we discuss and present the most significant publications concerning related problems in our approach of character animation.

1.3.1 Motion Synthesis

Designing a rich repertoire of realistic looking motions for virtual humans represents an important task for virtual entertainment software, i.e. computer games. Traditionally, animations are generated for virtual environments through *motion capture*, or dynamic simulation. Different methods for motion synthesis are presented in this section.

1.3.1.1 Kinematical Models based on Motion Capture

For the offline generation of highly realistic human movements in animated films and entertainment, motion capture (mocap) has become the standard approach. Thereby, the recorded complex movements of an actor can be transferred onto a virtual kinematic model, which results in realistic looking animations. Compared to traditional animation techniques such as keyframe animation, this method produces a larger amount of valuable data within a given time and convinces scientists to consider the motion capture system for animation [[BW95](#), [RGB96](#), [MBT99](#),

[BB07](#)]. In addition to this, adjustments can be made by editing or by blending the recorded movements in order to modify movement styles. Motion blending is a technique which combines multiple input motion data according to time-varying weights [[RCB98](#)]. It allows the user to generate either new parameterized motions or transitions by smoothly changing one animation into another one. With this method, hybrid motions can be synthesized and libraries of reusable motion clips can be created [[WP95](#)].

To make the synthesized movements compatible with constraints in a virtual world, retargetting techniques and kinematic boundary conditions have to be applied to adapt an animated motion from one character to another, with identical structure but with different segment lengths [[Gle98](#)]. In these cases, inverse kinematics, a common technique for positioning end effectors of articulated figures in individual frames of an animation, are used in addition to the motion capture data and helps to optimize the generation of the desired joint angle trajectories, if the standard approach does not yield satisfactory results for the special cases. These procedures work well and have been effectively used to produce motions for a number of applications and especially for commercial video games [[BCRP97](#)].

Furthermore, a large data base of captured motions is required to provide a sufficient set of body motions, when providing the ability to morph among different motions. The time to identify the requested motion from the large repertoire of movements is a critical factor for real-time applications. Recent approaches have tried to simplify this procedure by automatic selection and concatenation of recorded motion segments from large data bases, ensuring that the generated motion sequences fulfill constraints defined by the animator [[AFO03](#), [KG03](#), [SH07](#)]. Often a corpus of short motion capture clips is preprocessed that can be concatenated to generate a continuous motion sequence [[GSKJ03](#), [AF02](#)]. A simple graph structure facilitates efficient planning of character motions, where a user-guided process selects common character poses and system automatically synthesizes multi-way transitions that connect through these poses.

These methods permit a flexible off-line synthesis of quite complex sequences of movements. However, since the retrieval of the relevant trajectory segments from the database requires complex search algorithms with preprocessed motions, the resulting methods are typically not suitable for real-time generation of animations. In other words, although the mocap approach results in natural looking animations, it also requires tedious post-processing of the recorded motion capture data, which is almost impossible to transfer into real-time applications.

1.3.1.2 Physical Models

In many applications, such as in computer games or in robotics, a simulation in real-time is required, in which the characters are often based on physical or dynamical models. Therefore, truly interactive motion of the characters with the virtual environment is required. Characters have to react autonomously, taking the dynamic environment into account or respond to the actions of the applicant immediately. In addition to the real-time capability, the animated characters have to appear naturally, providing complex body movements with high degrees of realism, to get accepted as ‘realistic looking’ avatars by the skeptic audience. However, to obtain a realistic human motion the rigid body models must be physically correct, e.g. their mass and inertia properties are often derived from the biomechanics. Then, the desired movement can be performed applying control algorithms considering all physical constraints obtaining realistic models [TPB⁺89, TW90, HWBO95, GTH98, SHP04, CH07]. Hence, the control algorithms and control architecture become tedious and more complex, the more environmental features are considered.

In large crowd simulations, for instance, every individual character is treated as an autonomous virtual agent, who interacts with other individuals, e.g. by changing the heading or emotional state accordingly [MT01, UT02, GTH98, ST05]. Global path planning for each avatar quickly becomes computationally expensive, particularly in real-time contexts. Other approaches show crowd simulation without agent-based dynamics and rather unifies global path planning and local collision avoidance into a single optimization framework, where the agents do not experience a discrete regime change in the presence of other people, but perform global planning to avoid both obstacles and other agents [GKM⁺01, KS02, TCP06, ASDB08]. Therefore, the simulation is driven by dynamic potential fields, where the motion can be viewed as a perparticle energy minimization, and adopt a continuum perspective on the system. This formulation yields a set of dynamic potential and velocity fields over the domain that guide all individual motion simultaneously (Figure 1.3).

In general, physics based character and crowd simulation are a computationally intensive and expensive processes. Complex composite controllers have to be designed involving the solution to many equations. First, it is difficult to develop behavioral rules that consistently produce realistic motion iteratively; and second, global path planning for each agent quickly becomes computationally expensive, particularly in real-time contexts. For this reason, the underlying character models are often strongly simplified, resulting in a manageable complexity of the system dynamics and dynamics simulation, but lacking subtle details of realistic human body movements. In addition, physically based methods demand high expertise



Figure 1.3: Crowd animation. [TCP06]

of the animator to deal with a high parameter dimension, which is inappropriate for real-time animation systems.

1.3.1.3 Hybrid Models for Interactive Character Animation

In contrast to animation based on motion capture data, physical models allow virtual characters to interact dynamically with their surroundings. Because, the ability to simulate reactive responses to perturbations is essential and it is those natural responses to events that make the characters appear alive, the constructions of controllers for such physically based responses are difficult to realize (Section 1.3.1.2). However, researchers have succeeded in combining physics based models with motion-capture data, in order to generate realistic looking motion in real-time [SCCH09].

For the purpose of obtaining a natural motion, characters are animated using motion capture data, combined with dynamic simulations, constrained by biomechanical attributes for allowing responsive and interactive behavior. Zordan and Hodgins [ZH02, SCCH09] for example, proposed a method based on physics and motion capture data to simulate a ‘boxing scenario’, where specific gain parameters control the characters movement trajectories during a perturbation. While such perturbations may alter the trajectories depending on the physical force, characters return afterwards to an appropriate motion capture trajectory. They also used, a balance controller to keep the character upright while modifying sequences from a small motion library to accomplish specified tasks. Consequently, the system reacts to forces computed from a physical collision model by changing stiffness and damping terms.



Figure 1.4: A character switches automatically to obstacle avoidance mode and navigates around the obstacles [TLP07].

In order to expand the range of possible motions and to provide more natural interactions with the environment, researchers have focused on designing controllers for dynamically simulated characters. Given a corpus of motion capture data and a set of task dependent controllers, different motions can be obtained using blending methods in order to generate interactive motions. In addition environmental constraints like obstacles are taken into account and can be avoided in real-time [TLP07, dSAP08, MLPP09]. The underlying hierarchical control architectures are typically complex and require high expertise of the animator for the adjustment of their parameters [HPP05, CH05]. It seems interesting to develop simpler dynamical architectures that, however, can simulate complex human movements by integrating information learned by motion capture.

1.3.1.4 Dimensionality Reduction in Character Animation

Dimensionality reduction techniques allows an improved visualization, categorization, or simplification of large data sets. Several techniques for dimension reduction have been exercised in the context of human motion analysis for capturing the intrinsic dimensionality of the desired behavior and efficient computation. Motion capture sequences are often represented by large data sets due to the high sampling rate and the high number of degrees of freedom a virtual human body contains. Character motions are high dimensional but can be simplified by reducing the dimensionality of motion capture data in low-dimensional subspaces, using statistical methods. In this way an articulated figure motion can be optimized in the low-dimensional space to satisfy user-specified constraints, resulting in a data compression of motion data which economizes the memory.

One method for nonlinear dimensional reduction from motion capture data is based on Isomap and STIsomap to reduce the data set [SB05]. Jenkins and Mataric [JM04] focus on synthesizing humanoid motions from a motion database by automatically learning motion vocabularies. Starting with manually segmented motion capture data, ST-Isomap is applied to the motion segments in two passes, along with clustering techniques for each of the resulting sets of embeddings. Motion primitives and behaviors are then extracted and used for the motion synthesis.

Another typical approach is to apply the principal component analysis (Section 3) to compress and simplify the captured human motion data, and construct a low-dimensional motion space [AFO03, SHP04, SL06]. In this way, motion models can be constructed out of a rich set of model data and can be used for real-time applications [LT02]. Chai and Hodgins [CH05, CH07] developed an animation system that uses low-dimensional control signals obtained from the position of very few markers to animate characters. These control signals from the user's performance are supplemented by a database of pre-recorded full-body motion capture data. Therefore, local models are learned from a set of predefined motion capture examples, which are similar to the recorded marker locations. The information of these local models is then used to reconstruct the complete motion. Grochow et. al [GMHP04] applied a nonlinear dimensionality reduction technique called Gaussian Process Latent Variable Model (GPLVM) to motion data and then used the learned statistical model to compute poses from a small set of user-defined constraints. Thereby GPLVM allows the mapping from a low-dimensional space (latent space) to a feature space which characterizes motions e.g. joint angles. Hence, a kernel function maps the correlation between postures according to their corresponding representations in the latent space. The method generalizes RBF interpolation, providing an automatic learning of all RBF parameters.

1.3.2 The Idea of Synergies

1.3.2.1 Synergies

By increasing the complexity of human motion, the generation of articulated characters or humanoid robots becomes correspondingly harder since the number of degrees of freedom of an articulated character grows very rapidly with its complexity and makes the specification of the controls correspondingly extremely difficult. That is because every joint has its own number of degrees of freedom which make the human limb kinematically redundant.

The problem of motor redundancy can be illustrated using the following example [Lat08]: Touching the nose with a right index finger while moving the arm without losing contact shows the many possible combinations of arm-joint angles. Nevertheless, when the task is presented, we do it with a particular joint combination, where a particular joint configuration from an infinite number of possibilities has to be selected.

The assumption in motor control is that the Central Nervous System (CNS) organizes many degrees of freedom of the musculoskeletal system in order to generate a certain behavior [Ber67]. This complex task of organizing the desired behavior might be simplified by a hierarchical control architecture. This has led to the hy-

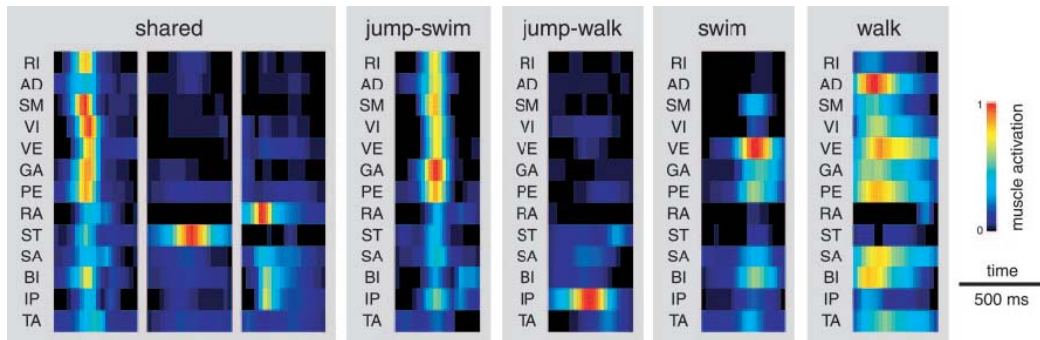


Figure 1.5: Shared synergies are extracted from the entire dataset whereas the other behavior-specific synergies are extracted from only the muscle patterns from two behaviors (jump-swim and jump-walk) or a single behavior (swim and walk)[dB05].

pothesis that the CNS uses elements (joints of a limb, muscles acting a joint) in a task specific way so that the complexity of this problem can be reduced into a set of *synergies* or *movement primitives*, as output modules [IPL04, FH05, TCd06].

In this manner, synergies can be specified as lower-dimensional control units that typically encompass only a subset of the available degrees of freedom. Decomposition in such low-dimensional sub-units has been proposed as a way to solve the ‘degrees-of-freedom problem’, which arises in the synthesis and control of movements in effector systems with many degrees of freedom. In other words, the crucial feature in this case is that different movements can be derived from a limited number of stored primitives by appropriate combination through a well defined syntax of actions to obtain a complex action [SIB03, FH05].

Studies in motor control have successfully applied unsupervised learning methods to extract low-dimensional spatio-temporal components from trajectories or EMG signals, which have been interpreted as correlates of synergies. [dSB03, dB05, IPL04, SFS98]. D’Avella and Bizzi [dB05] recorded electromyographic activity from different movements of frogs in naturalistic conditions. By using unsupervised learning methods, very few muscle synergies could be extracted, so that each behavior may be reconstructed by a combination of them (Figure 1.5). A similar approach was applied by recording EMG data of human locomotion and then extracting basic underlying components that can account a high level of the total variance across different muscles during normal gait [IPL04].

Based on the idea of synergies or movement primitives, we extended this approach to the field of character animation in order to generate different kinds of movements by mixing very few extracted components from human motion trajectories

in an appropriate way.

1.3.2.2 Humanoid Robots

Researchers in this field also seem inspired by the scientific field of motor control as they follow the idea of synergies or movement primitives, describing a subset of joints as well as their degrees of freedom, which were controlled by ‘Central Pattern Generators’ (CPGs) for the realization of periodic movements [LMIM09].

CPGs are neural circuits found in periodic movements of vertebrate and invertebrate animals, like swimming or walking that can produce rhythmic patterns of neural activity without receiving rhythmic inputs. They additionally present several interesting properties including distributed control, the ability to deal with redundancies, fast control loops, and allowing modulation of locomotion by simple control signals. These properties, when transferred to mathematical models, make CPGs interesting building blocks for locomotion controllers in robots [Jjs08].

CPGs have been proposed for the control of periodic motions of many kinds of different robots, for instance quadrupeds [CR94, KFHT02], bipeds [TYS91, RI06, MEN⁺06], and snake robots [CBGI05]. In biped robots, the artificial CPGs often consist of weakly coupled nonlinear dynamical systems, oscillators, that control the DOF of the joints, which are chain coupled in order to coordinate these several DOFs [RI06].

The numerous parameters of the CPGs, often specialized for a specific task, are usually set by automatic techniques like genetic algorithms, policy gradient or reinforcement learning [INS02, SIB03, GRIL08]. That makes the idea of the CPG much more efficient: The user is left with control of the resulting motion, which can be modified by changing the evaluation function. As opposed to this, the setting of each parameter manually leads to a full control over the system’s behavior but requires a lot of effort and time. However, CPGs encode rhythmic trajectories as limit cycles of nonlinear dynamical systems, typically systems of coupled nonlinear oscillators, which offer multiple interesting features. For instance, the stability properties (Section 6) of the limit cycle behavior (i.e. perturbations are quickly forgotten) or the smooth online modulation of trajectories through changes in the parameters of the dynamical system.

1.3.3 Dynamics for Collective Behavior

Dynamical systems derived, for example, from biomechanical or physical models seem particularly appropriate for real-time synthesis [TW90, GTH98, BRI06]. However, it has turned out that such models for the generation of human movements with high degree of realism typically have to be rather detailed [Ter09,

[HWBO95, AHS03], which results in complex dynamical systems whose properties are difficult to control. Consequently, the dynamical stability properties of such systems have rarely been addressed and it is an open question whether they can be treated at all, given their complexity. An important domain of the application of dynamical systems in computer animation is the simulation of autonomous and coordinated behavior of multiple characters. The dynamics of collective behaviors of animals have been extensively analyzed in biology for collective motions in flocks, as it is in the case of fish schools (Figure 1.6). Although each individual agent has no global knowledge about the group as a whole or about the surrounding environment, complex coordinated behaviors emerge from local interactions [Cou09, CDF⁺01].



Figure 1.6: Flocking example of a fish school: 'Tornado unter Wasser'. [HbF09]

One example for self-organized coordinated behavior is the tendency of multiple agents to synchronize their behavior, for instance during walking or applauding. From the scientific point of view, these observations of motion coordination pose novel challenges in system theory [PRK03] and serve as a remarkable tool for autonomous robotic vehicles and active sensors, where group coordination and cooperative control have been studied in the context of the navigation, or the control

of agents that realize coordinated behavior for useful tasks, e.g.[[Mat95](#), [BH97](#)].

Thereby, modeling of complex group behaviors for simulating autonomous characters is rarely discussed in the field of computer animation [[Rey99](#)]. Especially for large crowd scenes, the understanding of such coordinated organizations would help to permit the high computational costs, considering that each acting individual person has only access to local information about the behavior of the near-neighbors. In comparison, by using traditional computer animation techniques and scripting the path of a large number of individual characters, the computational costs become needless tedious to obtain a highly realistic crowd behavior [[MT01](#), [TWP03](#), [TCP06](#)].

Another characteristic and advantage of self-organized groups is that they are tolerant to loss or gain of group members, which makes them more robust to transient perturbations in contrast to non-interacting individuals [[CS07](#), [OEH02](#)]. This makes it interesting to exploit the underlying principles for the automatic synthesis of collective behavior in computer animation.

CHAPTER 2

Nonlinear Dynamical Systems

In nature, nonlinearity is the rule rather than the exception, while linearity is a simplification adopted for analysis. Therefore, nonlinear models are designed to provide a better mathematical way to characterize the inherent nonlinearity in real dynamic systems and may be expressed in the form of differential equations [CTF01, Str94]. In order to control such systems, detailed investigation of the systems' behavior and the circumstances of the stability properties have to attentively considered, since even small perturbations can lead to a chaotic and unpredictable behavior. Hence, the knowledge of the stability constraints is necessary to treat such systems in a controlled way to observe the desired behavior. Moreover, stable dynamic networks of nonlinear dynamical systems can be created.

This chapter mainly discusses the nonlinear-dynamical systems of limit-cycle oscillators: Van der Pol and Andronov-Hopf oscillators, which are applied in this thesis for generating periodic movements for character animation (Chapter 4). Moreover, this chapter focuses on the synchronization behavior of coupled oscillators. In addition to this, *contraction theory*, as a theoretical approach, permits a treatment of the dynamical properties of networks for different coupled nonlinear dynamical elements. Applying *contraction theory*, a method derived from system theory [WS05], stability properties for different kind of couplings between oscillators have been analyzed.

2.1 Synchronization

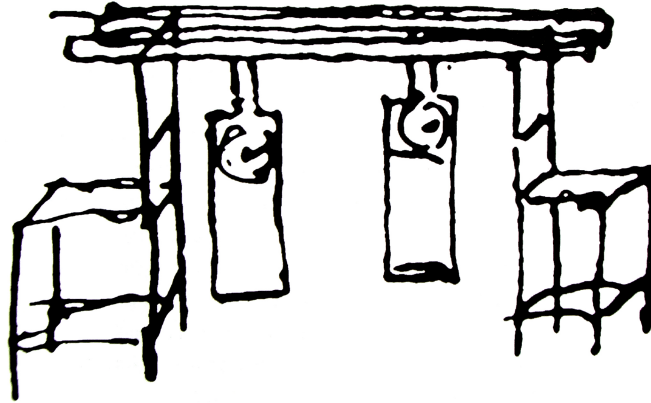


Figure 2.1: Two pendulum clocks coupled through a beam (Original picture from Huygen) [Huy67, PRK03].

Oscillators produce rhythmic outputs and play an important role in our environment; oscillatory processes can be observed in our surroundings, like in nature or in social life (e.g. applauding, heart beat, butterflies flapping their wings, pendulum of a clock, violins in an orchestra, etc.).

Synchronization refers to the phenomena of coordinated behavior of events in rhythm and can be understood as an adjustment of rhythms of oscillating objects due to their weak interaction [PRK03].

Fireflies, in Southeast Asia, provide a great example of a synchronization behavior in nature. A large group of male fireflies gather in neighboring trees and synchronize their light flashes at night in order to attract the female fireflies passing by. Although the fireflies start flashing with independent intervals, they seem to influence each other by adapting or regulating the speed of the flashing and synchronizing their light emissions very precisely during the night.

In 1665, Christiaan Huygens, a Dutch researcher in mathematics, astronomy, and physics, came across the synchronization phenomena in pendulum clocks. He recognized that a couple of pendula of clocks, which were hanging on the wall, had synchronized when the pendula moved always in the opposite direction. In a letter to his father, he described his discovery [Huy67, PRK03]:

'It is quite worth noting that when we suspended two clocks so constructed from two hooks embedded in the same wooden beam, the motions of each pendula in opposite swings were so much in agreement that they never receded the least bit

from each other and the sound of each was always heard simultaneously. Further, if this agreement was disturbed by some interference, it reestablished itself in a short time....The cause is that the oscillations of the pendula, in proportion to the weight, communicate some motion to the clocks. This motion, impressed onto the beam, necessarily has the effect of making the pendula come to a state of exactly contrary swings if it happened that they moved otherwise at first, and from this finally the motion of the beam completely ceases.'

Thus, Huygens describes the synchronization behavior of the two pendulum clocks due to the coupling through the beam, where the pendula of the clocks can be viewed as self-sustained oscillators (Figure 2.1).

2.1.1 Self-Sustained Oscillator

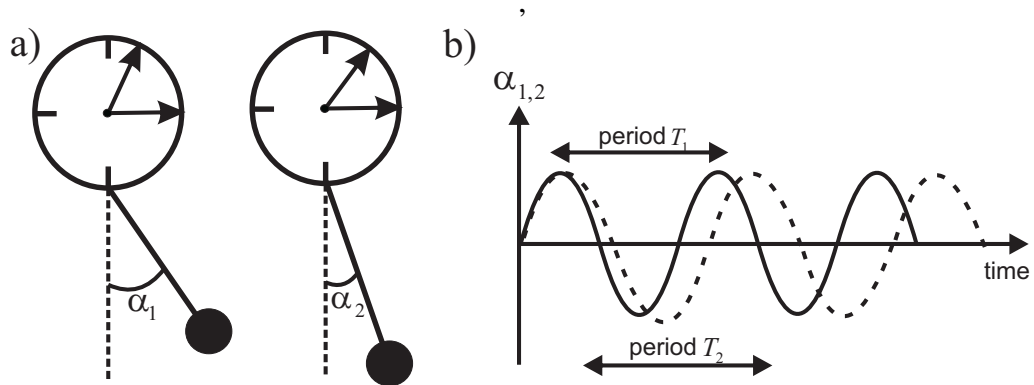


Figure 2.2: Two non-identical pedulum clocks: a) The pendula are in different positions $\alpha_1 \neq \alpha_2$ at some arbitrary moment of time. b) They differ in their periods $T_2 > T_1$.

Self-Sustained oscillators are autonomous and can be described within a class of nonlinear dynamics, where the shape of the oscillation is dependent on the parameters of the system. One of the main properties states that even by taking the systems apart from each other, the system keeps generating the oscillatory outputs in its own rhythm due to its own source of energy. In addition, the oscillators are characterized by their stability or robustness against, at least small, perturbation. After the system gets disturbed, following some transient process, the shape of the oscillators converge to their nominal motion and the disturbance is somehow 'forgotten'.

Rhythm and frequency: The rhythm can be formalized by the number of oscillation cycles per time unit or period, or by the oscillation cyclic frequency: $f = \frac{1}{T}$

where the period, denoted by T , describes the length of one cycle.

In theoretical conventions, the angular frequency (or angular velocity) $\omega = 2\pi f = 2\pi/T$ is a more common representation and can be viewed as a scalar measure of the rotation rate.

To describe the basic synchronization behavior of two self-sustained oscillators, it is assumed that two non-identical pendulum clocks, with different oscillation periods (Figure 2.2), are hanging on a common support and are coupled in a way that they interact through the vibration of the beam.

Synchronization in anti-phase (Figure 2.3): As it was mentioned before in Section 2.1, Huygens described a synchronization behavior of the pendulum clocks in anti-phase, where the pendula of two clocks move in opposite directions so that the phase difference is given by: $\phi_2 - \phi_1 \approx \pi$.

Synchronization in phase (Figure 2.3): Two non-identical oscillators with their own frequency are coupled together. They start adjusting their rhythms with each other, and end up with a common frequency. The coincidence of the frequencies is often called *frequency entrainment* or *locking*. Hence, the phase difference is defined by: $\phi_2 - \phi_1 = 0$. Concerning the pendulum example, it means that the pendula move exactly in the same direction and swing equally to the other direction.

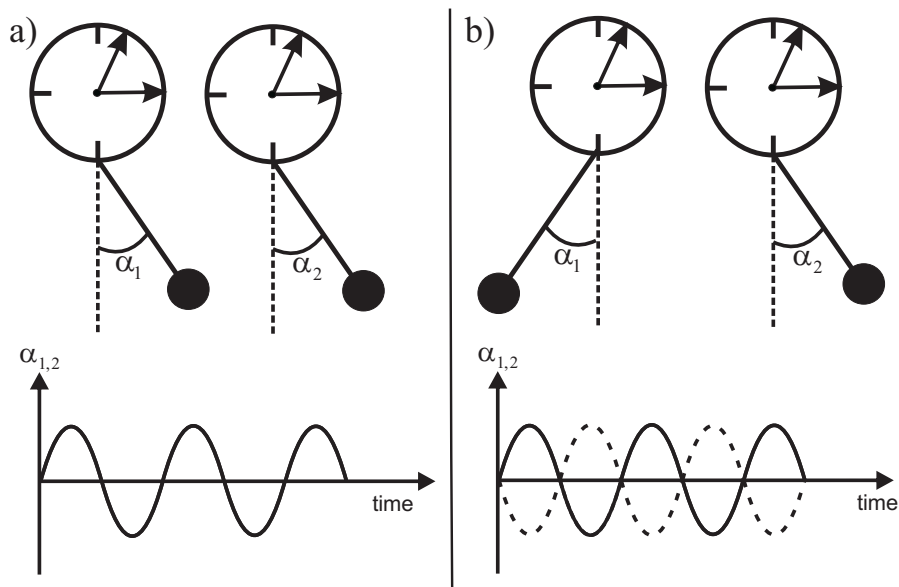


Figure 2.3: a) Synchronization in phase. b) Synchronization in anti-phase.

Phase-shift: In the case of two pendulum clocks, starting with initially different oscillation periods, phase shifts can be observed and the pendula denote different positions $\alpha_1 \neq \alpha_2$ at time t .

However, whether they synchronize or not depends on the coupling strength, which define how weak or how strong the interaction is.

2.2 Limit Cycle Oscillators

2.2.1 Phase Space

Generally speaking, a phase space is a representation, in which all possible states of the system are displayed. In other words, every degree of freedom or parameter of the system is embodied in each point in the phase space. The phase-space of i.e. limit cycle oscillator illustrates two degrees of freedom –position and velocity– and portrait the dynamics of this system. Consider a dynamical system with f_1 and f_2 given functions:

$$\dot{x}_1 = f_1(x_1, x_2) \quad \text{and} \quad \dot{x}_2 = f_2(x_1, x_2) \quad (2.1)$$

Equation 2.1 can be rewritten in a more compact vector notation:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$$

with $\mathbf{x} = (x_1, x_2)$ and $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}))$. Then, \mathbf{x} represents a point in the phase plane, and $\dot{\mathbf{x}}$ is the velocity vector at that point.

2.2.2 Limit Cycle

A solution trajectory, $(x_1(t), x_2(t))$ is defined as a closed orbit, or periodic solution, if for some time $t = T$: $(x(t+T), y(t+T)) = (x(t), y(t))$, for all $t \geq 0$. The limit cycle on a phase-plane is defined by an isolated closed trajectory or closed orbit, whereby the neighboring trajectories of the dynamical system either spiral towards or away from the limit cycle, as illustrated in Figure 2.4. If all the neighboring trajectories approach the limit-cycle, it is called stable or attracting [Str94]. Therefore, the essential characteristics of limit cycle oscillators, representing non-linear autonomous dynamic systems, imply self-sustained oscillations, having the property that the trajectories converge against the limit cycle independently of their initial conditions.

Additionally, any small perturbation causes the system to return or relax the disturbed trajectories, after a transient time, back towards the limit-cycle.

The Van der Pol and Andronov-Hopf oscillators represent the two most common

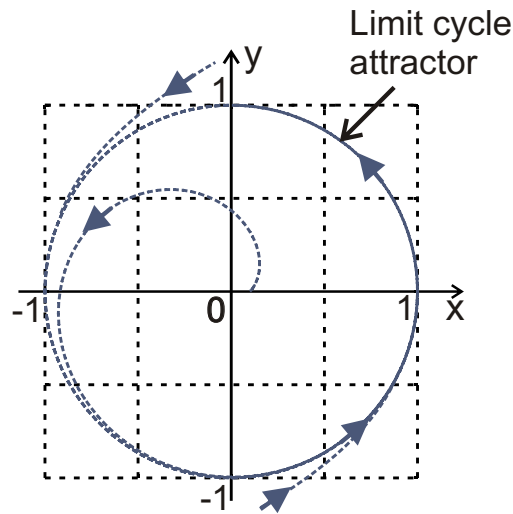


Figure 2.4: Limit-cycle oscillator

limit-cycle oscillators and will be introduced in the next Section.

2.2.3 Van der Pol Oscillator

The Van der Pol oscillator was proposed in 1929 by the Dutch engineer Balthasar Van der Pol and can be understood as a harmonic oscillator with amplitude-dependent damping. The dynamics of this limit-cycle oscillator is given by the following differential equation:

$$\ddot{y}(t) + \underbrace{\zeta(y(t)^2 - a)}_{\text{Amplitude-dependent damping term}} \dot{y}(t) + \omega_0^2 y(t) = 0 \quad (2.2)$$

where ω_0 determines the eigenfrequency and a the amplitude of the stable limit cycle. The nature of the limit cycle is dependent on the value $\zeta > 0$ of the amplitude-dependent damping term: $\zeta(y(t)^2 - a)$. For $\zeta = 0$ the dynamics represents a harmonic oscillator, where the oscillatory motion is sinusoidal, since the damping term vanishes completely:

$$\ddot{y}(t) + \omega_0^2 y(t) = 0$$

When $\zeta > 0$, the state will enter the limit cycle, where energy continues to be conserved. In addition, for large values of y , the damping term becomes positive and the damper removes energy from the system, which implies a convergent

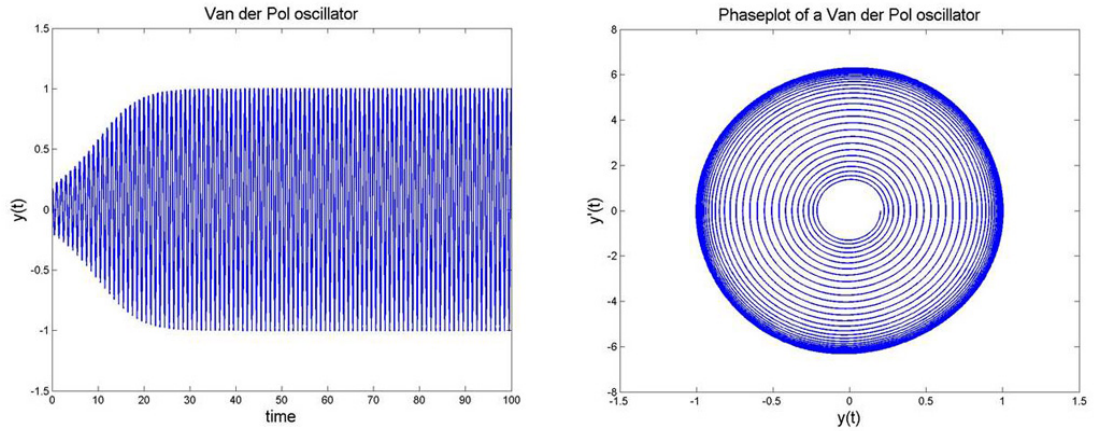


Figure 2.5: Van der Pol oscillator viewed in the time domain (left) and in the phase space (right).

motion towards the limit cycle. Conversely, for small values of y , the damping term becomes negative and the damper adds energy into the system, in order to approach to the attractor. Since nonlinear damping varies with y to adjust the amplitude, the system motion can neither grow unboundedly nor decay to zero. The dynamical system can be rewritten in two first order systems, where

$$\dot{y}(t) = x(t)$$

then

$$\dot{x}(t) = \zeta(a - y(t)^2) x(t) - \omega_0^2 y(t)$$

Figure 2.5 illustrates the Van der Pol oscillator as a function of time (left) and in phase space (velocity \dot{y} against position y) on the right side. The typical behavior of a limit cycle oscillator (Section 2.2) is shown clearly: Independent of any initial condition, the convergence towards a stable equilibrium amplitude can be obtained and small perturbations are ‘forgotten’.

2.2.4 Andronov-Hopf Oscillator

The Andronov-Hopf oscillator, a nonlinear oscillator, is characterized by a limit cycle that corresponds to a circular trajectory in phase space. The dynamics of the Andronov-Hopf oscillator is described by two first order differential equations [AVK87]:

$$\begin{aligned} \dot{x}(t) &= \zeta(a - (x^2(t) + y^2(t))) x(t) - \omega y(t) \\ \dot{y}(t) &= \zeta(a - (x^2(t) + y^2(t))) y(t) + \omega x(t) \end{aligned} \quad (2.3)$$

Similar to the Van der Pol Oscillator, the parameters ω , a and ζ describe the eigenfrequency, the amplitude, and the speed control. The amplitude-dependent damping term is determined by $\zeta((x^2(t) + y^2(t)) - a)$. Whereby, the parameters influence the system in the same way as described in Section 2.2.3.

Introducing polar coordinates $r = \sqrt{x^2 + y^2}$ and $\phi = \arctan(y/x)$, this system can be rewritten:

$$\begin{cases} \dot{r}(t) = r(t)(1 - r^2(t)) \\ \dot{\phi}(t) = \omega \end{cases} \quad (2.4)$$

The polar transformation has produced a pair of decoupled differential equation, hence, the first equation does not depend on ϕ anymore and conversely ϕ does not depend on r . Indeed, the subsystem has three stationary solutions where $f(r) = 0$. It has fixed points at $r_1 = 0$ corresponding to $(x, y) = (0, 0)$ (unstable) and $r_2 = \pm 1$ (stable) corresponding to the unit limit cycle (see also Section 2.4.1).

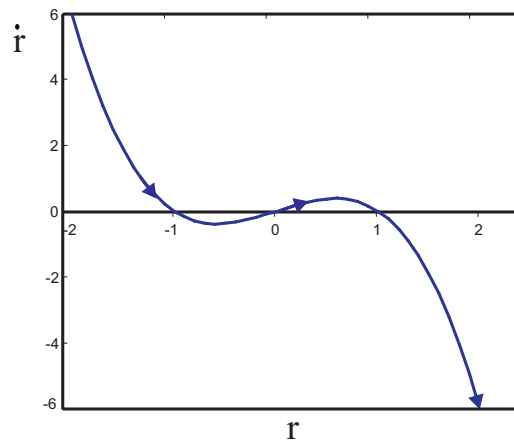


Figure 2.6: The radial dynamics has three stationary solutions where $\dot{r} = f(r) = 0$. Whereas the fixed point at $r_1 = 0$ is unstable, the other two $r_2 = \pm 1$ are stable.

2.3 Contraction Theory and Partial Contraction Theory

Contraction theory [LS98] provides a general method for the analysis of essentially nonlinear systems and moreover, it can be applied for the analysis of more complex systems comprising many components and has already been applied successfully in different types of systems [LS98, Slo03, WS05, PS07].

The classical approach for the stability analysis of nonlinear systems is to compute first the stationary solutions of the dynamics, and then to establish its *local stability* by linearization in the neighborhood of this solution numerically.

Contraction theory takes a different approach and characterizes the system stability by the behavior of the differences between solutions with different initial conditions. If these differences vanish exponentially over time, and its solution converges towards a single trajectory independently from the initial states, the system is called *globally asymptotically stable*. Interestingly, the analysis of such differences between solutions is often simpler than the classical linearization approach, making those systems tractable that would be impossible to analyze with the classical approach. The contraction theory [LS98] can be summarized in the following features:

1. Global exponential convergence and stability are guaranteed.
2. Asymptotic convergence rates can be explicitly computed as eigenvalues.
3. Robustness to variations in dynamics can be easily quantified.
4. Under simple conditions, convergence to a synchronized state can be preserved through system combinations (if every subsystem has contracting property, the contracting of the global system can always be guaranteed by an appropriate choice of couplings).

2.3.1 Contraction Theory

In the following section the bounds are derived for the convergence properties of the solution of a dynamical system. Consider a nonlinear dynamical system:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, t) \quad (2.5)$$

The bounds for the convergence property depend on the eigenvalues of the symmetrized *Jacobian* of the system:

$$\mathbf{J}_s(\mathbf{x}, t) = \left(\frac{\partial \mathbf{f}(\mathbf{x}, t)}{\partial \mathbf{x}} \right)_s \quad (2.6)$$

Given a square matrix \mathbf{A} , the matrix

$$\mathbf{A}_s = (\mathbf{A} + \mathbf{A}^T)/2 \quad (2.7)$$

signifies its symmetric part. Moreover, we denote the real-valued matrix functions $\lambda_{\min}(\mathbf{A}_s)$ and $\lambda_{\max}(\mathbf{A}_s)$ which correspond to the smallest, respectively largest,

eigenvalue of the symmetric matrix \mathbf{A}_s . The matrix \mathbf{A}_s is **positive definite**, denoted by $\mathbf{A}_s > \mathbf{0}$ if

$$\lambda_{\min}(\mathbf{A}_s) > 0$$

and **negative definite**, denoted by $\mathbf{A}_s < \mathbf{0}$ if

$$\lambda_{\max}(\mathbf{A}_s) < 0.$$

The matrix inequality $\mathbf{A} > \mathbf{B}$ denotes $\lambda_{\min}(\mathbf{A}_s) > \lambda_{\max}(\mathbf{B}_s)$. If the matrix itself is a function of state and time (i.e., $\mathbf{A}_s(\mathbf{x}, t)$) one says that it is **uniformly positive definite** if there exists a real $\iota > 0$ such that

$$\forall \mathbf{x}, \forall t : \lambda_{\min}(\mathbf{A}_s(\mathbf{x}, t)) \geq \iota$$

Likewise, it is defined as **uniformly negative definite** if there exists a $\iota > 0$ such that

$$\forall \mathbf{x}, \forall t : \lambda_{\max}(\mathbf{A}_s(\mathbf{x}, t)) \leq -\iota$$

2.3.2 Contraction of Dynamical Systems

Assume that $\mathbf{x}(t)$ is one solution of the dynamical system 2.5 and

$$\tilde{\mathbf{x}}(t) = \mathbf{x}(t) + \delta\mathbf{x}(t) \tag{2.8}$$

a neighboring one. The function $\delta\mathbf{x}(t)$ is also called *virtual displacement* (see Figure 2.7). If the virtual displacement is small enough, the last equation 2.8, together with equation 2.5 indicates

$$\frac{d}{dt}\delta\mathbf{x}(t) = \mathbf{J}(\mathbf{x}, t)\delta\mathbf{x}(t)$$

implying through

$$\frac{d}{dt}\|\delta\mathbf{x}(t)\|^2 = \delta\mathbf{x}^T(t) \delta\dot{\mathbf{x}}(t) = 2\delta\mathbf{x}^T(t)\mathbf{J}_s(\mathbf{x}, t)\delta\mathbf{x}(t)$$

Denoting that $\lambda_{\max}(\mathbf{J}_s(\mathbf{x}, t))$ presents the largest eigenvalue of the symmetric part of the Jacobian $\mathbf{J} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}}$; it follows the inequality:

$$\|\delta\mathbf{x}(t)\| \leq \|\delta\mathbf{x}(0)\| e^{\int_0^t \lambda_{\max}(\mathbf{J}_s(\mathbf{x}, s)) ds} \leq \|\delta\mathbf{x}(0)\| e^{-\rho_c t}$$

If the Jacobian is uniformly negative definite $\mathbf{J}_s(\mathbf{x}, t) < \mathbf{0}$ then $\lambda_{\max}(\mathbf{J}_s(\mathbf{x}, t))$ is the maximal negative eigenvalue. Thus, this equation implies that any nonzero

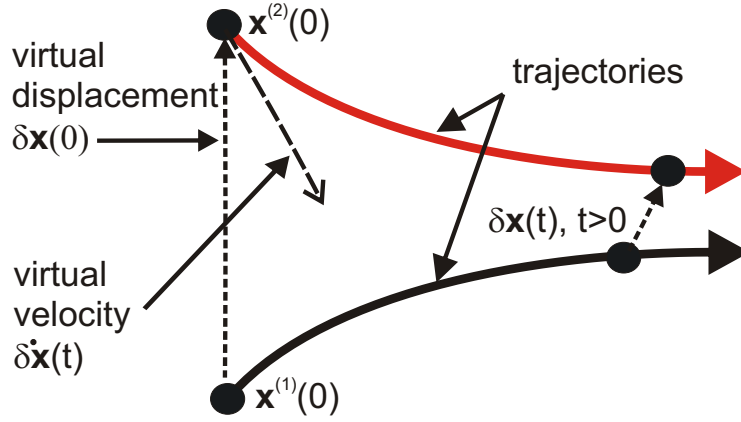


Figure 2.7: Two trajectories of a dynamical system and the virtual displacement.

virtual displacement $\|\delta\mathbf{x}\|$ decays exponentially to zero over time. This decay occurs with a *convergence rate* (inverse timescale) that is bounded by the quantity

$$\rho_c = -\sup_{\mathbf{x}, t} \lambda_{\max}(\mathbf{J}_s(\mathbf{x}, t)).$$

By ‘concatenating’ such virtual displacements at fixed points in time one can show that any difference between the trajectories decays to zero, with at least the following time constant [LS98]. This has the consequence that all trajectories converge towards a single trajectory exponentially. Therefore, this motivates:

Definition 1 *With respect to the dynamical system $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, t)$, the regions in state space for which the symmetrized Jacobian*

$$\mathbf{J}_s(\mathbf{x}, t) = \frac{1}{2} \left[\frac{\partial \mathbf{f}(\mathbf{x}, t)}{\partial \mathbf{x}} + \left(\frac{\partial \mathbf{f}(\mathbf{x}, t)}{\partial \mathbf{x}} \right)^T \right] < \mathbf{0}$$

is uniformly negative definite are called contracting regions. All solutions that start in these regions converge towards a single trajectory for $t \rightarrow \infty$.

The previous argumentation can be extended by measuring the length of the virtual displacement using a different metric (coordinate system). Assume a coordinate transformation:

$$\delta\mathbf{z} = \Theta\delta\mathbf{x}$$

$\Theta(\mathbf{x}, t)$ presents a uniformly invertible square matrix, which in most cases is state- and time-dependent. One can introduce the transformed displacement

$$\delta\mathbf{z}(t) = \Theta(\mathbf{x}, t)\delta\mathbf{x}(t)$$

Analogous to the previous case, one finds:

$$\frac{d}{dt}(\delta \mathbf{z}^T \delta \mathbf{z}) = 2\delta \mathbf{z}^T \dot{\delta \mathbf{z}} = 2\delta \mathbf{z}^T \underbrace{\left(\dot{\Theta} + \Theta \frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)}_{\mathbf{F}} \Theta^{-1} \delta \mathbf{z}$$

This implies the following general result [LS98]:

Theorem 1 *Assume that for the system 2.5 it is possible to find a square matrix $\Theta(\mathbf{x}, t)$ such that $\Theta(\mathbf{x}, t)^T \Theta(\mathbf{x}, t)$ is uniformly positive definite, and such that the generalized Jacobian*

$$\mathbf{F} = \left(\dot{\Theta} + \Theta \frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right) \Theta^{-1} \quad (2.9)$$

is uniformly negative definite, then all system trajectories converge exponentially to a single trajectory, and the system is called contracting. The rate of convergence of $\|\delta \mathbf{z}(t)\|$ is at least $\rho_c = -\sup_{\mathbf{x}, t} \lambda_{\max}(\mathbf{F}_s(\mathbf{x}, t))$.

The matrix $\mathbf{M}(\mathbf{x}, t) = \Theta(\mathbf{x}, t)^T \Theta(\mathbf{x}, t)$ is also called the contraction metric.

Conversely, the existence of a uniformly positive definite metric

$$\mathbf{M}(\mathbf{x}, t) = \Theta(\mathbf{x}, t)^T \Theta(\mathbf{x}, t)$$

with respect to which the system is contracting is a necessary condition for the global exponential convergence of trajectories [LS98]. Furthermore, all transformations Θ corresponding to the same \mathbf{M} result in the same eigenvalues for the symmetric part of \mathbf{F} [Slo03], and thus the same convergence rate. (The proofs can be found in [LS98, Slo03].)

2.3.3 Partial Contraction

The concept of partial contraction can be seen as an extension of contraction theory. The application implies the analysis of convergence to behaviors or to specific properties, such as equality of state components, or convergence to a manifold, rather than trajectories. Many systems are not contracting with respect to all dimensions of the state space, but show convergence with respect to a subset of dimensions. A typical example is an externally driven nonlinear oscillator. By its tendency to self-initiate oscillatory solutions, it is unstable and thus non-contracting, within a region around the origin of state space. However, independent of the initial state, it might converge exponentially against a single trajectory that is determined by the external driving signal.

Partial contraction [WS05] allows to capture this property in a mathematically

well-defined manner, and to derive from it results in the global stability of the system. The key idea is to construct an auxiliary system that is contracting with respect to a subset of dimensions (or submanifold) in state space.

Theorem 2 *Consider a nonlinear system of the form*

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{x}, t) \quad (2.10)$$

and assume that the auxiliary system

$$\dot{\mathbf{y}} = \mathbf{f}(\mathbf{y}, \mathbf{x}, t) \quad (2.11)$$

is contracting with respect to \mathbf{y} uniformly for all relevant \mathbf{x} . If a particular solution of the auxiliary system verifies a specific ‘smooth property’, then all trajectories of the original system 2.10 verify this property with exponential convergence. The original system is then said to be partially contracting.

A ‘smooth property’ is a property of the solution that depends smoothly on space and time, such as convergence against a particular solution or value.

The proof of the theorem is to immediately notice that the observer-like system 2.11 has $\mathbf{y}(t) = \mathbf{x}(t)$ for all $t \geq 0$ as a particular solution. Since all trajectories of the \mathbf{y} -system converge exponentially to a single trajectory, this implies that also the trajectory $\mathbf{x}(t)$ verifies this specific property with exponential convergence.

Related to partial contraction are the following methods that will be crucial for the derivation of results for the synchronization of groups of avatars. Again starting from the equation 2.5, we assume the existence of a *flow-invariant linear subspace* \mathcal{M} , i.e. a linear subspace \mathcal{M} such that

$$\forall t : \mathbf{f}(\mathcal{M}, t) \subset \mathcal{M}$$

(see also Figure 2.8).

This implies that any trajectory starting in \mathcal{M} remains in \mathcal{M} . Furthermore, we assume that $p = \dim(\mathcal{M})$ and consider an orthonormal basis $(\mathbf{e}_1, \dots, \mathbf{e}_n)$ where the first p vectors form a basis of \mathcal{M} and the last $n - p$ a basis of \mathcal{M}^\perp , the orthogonal space of \mathcal{M} . We define an $(n - p) \times n$ matrix \mathbf{V} whose rows are $\mathbf{e}_{p+1}^T, \dots, \mathbf{e}_n^T$. This matrix can be regarded as projection on \mathcal{M}^\perp , which implies $\mathbf{x} \in \mathcal{M} \Leftrightarrow \mathbf{V}\mathbf{x} = 0$. It verifies

$$\mathbf{V}\mathbf{V}^T = \mathbf{I}_{n-p} \quad \text{and} \quad \mathbf{V}^T\mathbf{V} + \mathbf{U}^T\mathbf{U} = \mathbf{I}_n$$

where \mathbf{U} is the matrix formed by the first p basis vectors.

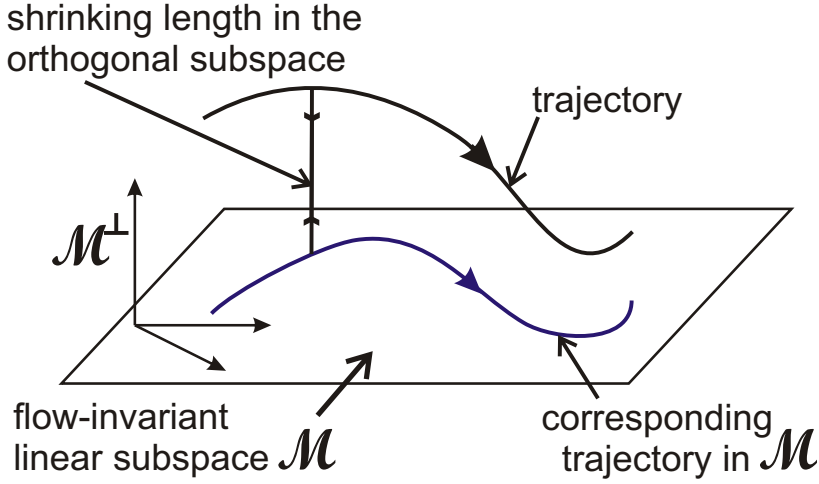


Figure 2.8: Flow-invariant *linear subspace* \mathcal{M} , such that $\forall t : f(\mathcal{M}, t) \subset \mathcal{M}$. Any trajectory starting in \mathcal{M} remains in \mathcal{M} .

Theorem 3 Assuming that for the dynamical system 2.5, a flow-invariant linear subspace \mathcal{M} exists with the associated orthonormal projection matrix \mathbf{V} . A particular solution $\mathbf{x}_p(t)$ of this system converges exponentially to \mathcal{M} if the auxiliary system

$$\dot{\mathbf{y}} = \mathbf{V}\mathbf{f}(\mathbf{V}^T\mathbf{y} + \mathbf{U}^T\mathbf{U}\mathbf{x}_p(t), t) \quad (2.12)$$

is contracting with respect to \mathbf{y} for all relevant \mathbf{x}_p , then starting from any initial conditions, all trajectories of the original system will exponentially converge to the invariant subspace \mathcal{M} . Furthermore, if all the contraction rates for the system 2.12 are lower-bounded by some constant $\lambda > 0$, uniformly in \mathbf{x}_p and in a common metric, then the convergence to \mathcal{M} will be exponential with a minimum rate λ .

The proof for this theorem can be found in [PS07]. It implies that a simple sufficient condition for global exponential convergence to \mathcal{M} is given by the following inequality that needs to hold uniformly:

$$\mathbf{V} \left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right)_s \mathbf{V}^T < \mathbf{0}$$

An even more general condition can be derived if there exists a constant invertible transform Θ on \mathcal{M}^\perp such that

$$\Theta \mathbf{V} \left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right)_s \mathbf{V}^T \Theta^{-1} < \mathbf{0}$$

is fulfilled uniformly [PS07].

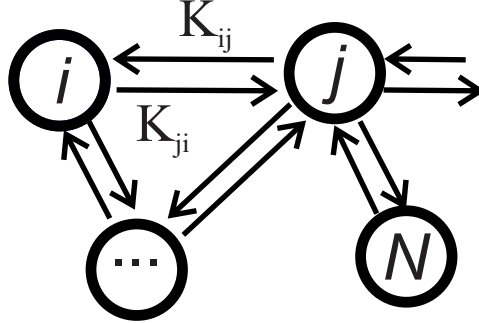


Figure 2.9: Diffusive coupling with equal coupling gains K , where \mathcal{N}_i define a set of neighbors of i .

2.4 Contraction Analysis for Coupled Nonlinear Dynamical Systems

Most coupled oscillators in natural world are networked in large groups, such as fireflies, with synchronized flashes, partial contraction, which was described previously in Sections 2.3.2, 2.3.3 can be used to study the synchronization behavior. This technique can be applied to analyze the stability for networks of coupled dynamical elements of various coupling structures. Based on observed stability properties, a synchronization behavior of the dynamical primitives can be achieved. This approach has been used to analyze the dynamics in this thesis, to analyze the stability of crowd animation, whose characters are coupled in an appropriate way to obtain a coordinate behavior which will be discussed later in more detail (see Chapter 6). Assume in the following n dynamical systems with a linear, very general coupling structure, so called *diffusive coupling* [WS05], where K_{ij} denotes the gain (associated with coupling from node i to j). Furthermore, assume that coupling links are bidirectional and symmetric in different directions, i.e., $K_{ij} = K_{ji}$, illustrated in Figure 2.9:

$$\dot{\mathbf{x}}_i = \mathbf{f}(\mathbf{x}_i, t) + \sum_{j \in \mathcal{N}_i} \mathbf{K}_{ij}(\mathbf{x}_j - \mathbf{x}_i) \quad \forall i = 1, \dots, n \quad (2.13)$$

where \mathcal{N}_i denotes the set of indices of the active links of element i . In this case of diffusive coupling, after synchronization, the dynamics of each subsystem is equivalent to the dynamics of uncoupled subsystem.

The matrix \mathbf{L} with the blocks ($\mathbf{L}_{ii} = \sum_{j \in \mathcal{N}_i} \mathbf{K}_{ij}$ and $\mathbf{L}_{ij} = -\mathbf{K}_{ij}$ for $j \neq i$) is

called *Laplacian matrix* of the coupling graph. With this matrix and the definitions

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_n \end{bmatrix}, \quad \mathbf{f}(\mathbf{x}, t) = \begin{bmatrix} \mathbf{f}(\mathbf{x}_1, t) \\ \vdots \\ \mathbf{f}(\mathbf{x}_n, t) \end{bmatrix}$$

the equation system can be written in vector form:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, t) - \mathbf{L}\mathbf{x}$$

This implies that the Jacobian of the system is given by $\mathbf{J}(\mathbf{x}, t) = \mathbf{D}(\mathbf{x}, t) - \mathbf{L}$, where

$$\mathbf{D}(\mathbf{x}, t) = \begin{bmatrix} \frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\mathbf{x}_1, t) & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\mathbf{x}_n, t) \end{bmatrix} \quad (2.14)$$

For the case of diffusive coupling, we assume again the existence of a flow-invariant linear subspace \mathcal{M} of the \mathbf{x} space that contains a particular solution of the form $\mathbf{x}_1^* = \dots = \mathbf{x}_n^*$.

$$\mathbf{f}(\mathbf{x}^*, t) - \mathbf{L}\mathbf{x}^* = \begin{bmatrix} \mathbf{f}(\mathbf{x}_1^*, t) \\ \vdots \\ \mathbf{f}(\mathbf{x}_n^*, t) \end{bmatrix} \in \mathcal{M}$$

For this solution all state variables \mathbf{x}_i for different subsystems i are identical and thus in synchrony. In addition, for this solution the coupling term in equation 2.13 vanishes so that the form of the solution is identical to the solution of the uncoupled systems $\dot{\mathbf{x}}_i = \mathbf{f}(\mathbf{x}_i, t)$.

From the last section, it can be implied that \mathbf{V} is a projection matrix onto the subspace \mathcal{M}^\perp , a sufficient condition for convergence toward this solution, which is the matrix inequality $\mathbf{V}(\mathbf{D}(\mathbf{x}, t) - k\mathbf{L})_s \mathbf{V}^T < \mathbf{0}$. From this inequality, the following sufficient condition for exponential convergence can be derived [PS07]

$$\lambda_{\min}(\mathbf{V}k\mathbf{L}_s \mathbf{V}^T) > \sup_{\mathbf{x}, t} \lambda_{\max} \left(\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}(\mathbf{x}, t) \right)_s \right) \quad (2.15)$$

which implies the following minimum convergence rate:

$$\rho_c = - \sup_{\mathbf{x}, t} \lambda_{\max}(\mathbf{V}(\mathbf{D}(\mathbf{x}, t) - k\mathbf{L})_s \mathbf{V}^T)$$

2.4.1 Andronov-Hopf Oscillator

The methods from contraction theory can be exploited to analyze the dynamics of networks of coupled non-linear dynamical systems. Therefore, the stability properties of oscillators (limit-cycle oscillator, viewed earlier in Section 2.1.1 and networks of such interacting systems will be discussed in more detail. Providing mathematical results help us to design a stable collective behavior as e.g. in crowds (Section 6).

For this purpose, the dynamics of an individual character is modeled by an Andronov-Hopf oscillator, a nonlinear oscillator whose choice of parameters is characterized by a limit cycle that corresponds to a circular trajectory in phase space (Section 2.2.4). For appropriate re-parameterization (rescaling of time and state-space axes) the dynamics of this oscillator is described by the differential equations [AVK87]:

$$\begin{cases} \dot{x}(t) = (1 - (x^2(t) + y^2(t))) x(t) - \omega y(t) \\ \dot{y}(t) = (1 - (x^2(t) + y^2(t))) y(t) + \omega x(t) \end{cases} \quad (2.16)$$

which can be written more compact in vector form (with $\mathbf{x} = [x, y]^T$):

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}, t) \quad (2.17)$$

In the following equations, we will omit the notation of time dependence for the oscillators phase variables: $r(t)$, $\phi(t)$, $x(t)$, $y(t)$. The symmetrized Jacobian for the system in \mathbf{x} coordinates is:

$$\mathbf{J}_s = \frac{1}{2} \left(\frac{\partial \mathbf{f}^T}{\partial \mathbf{x}} + \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right) = \begin{bmatrix} -(x^2 + y^2 - 1) - 2x^2 & -2xy \\ -2xy & -(x^2 + y^2 - 1) - 2y^2 \end{bmatrix}$$

The eigenvalues of this matrix $\lambda_1 = 1 - (x^2 + y^2)$ and $\lambda_2 = 1 - 3(x^2 + y^2)$ are independent of ω . The contracting region for the system is $(x^2 + y^2) > 1$, where all trajectories converge to the circular trajectory of the limit cycle $r^2 = 1$.

(Inside the disk $r^2 < 1$ a single unstable fixpoint $\mathbf{x} = \mathbf{0}$ exists.) Due to the Cauchy theorem the solutions of this ordinary differential equation (ODE) do not cross each other. Due to topological arguments for the 2D autonomous system with a single unstable point in a compact region with single attractor trajectory as its border, we have the global convergence of all trajectories of Hopf oscillator to the stable limit cycle.

By rewriting the system using polar coordinates (see equation 2.4) the symmetrized Jacobian of this system is given by

$$\mathbf{J}_s = \begin{bmatrix} 1 - 3r^2 & 0 \\ 0 & 0 \end{bmatrix}$$

showing that, according to Definition 1, this system is semi-contracting [PS07] in the region $|r| > 1/\sqrt{3}$ where its symmetrized Jacobian is uniformly negative definite. Meanwhile, a more general result can be obtained by using a different metric (cf. Section 2.4.1.1). The introduction of the new variable $\rho = 1/r^2 > 0$ transforms the dynamics into the form:

$$\begin{aligned} (\dot{r}^2) &= 2r^2(1 - r^2) \\ \Rightarrow \dot{\rho} &= 2(1 - \rho) \end{aligned} \tag{2.18}$$

In this case, one of the eigenvalues of the symmetrized Jacobian are -1 and 0 , so that the system is semi-contracting in the whole phase plane, excluding the points with $\rho = 0$.

2.4.1.1 Symmetric Coupling of Two Oscillators

The constraints that guarantee the synchronization of two symmetrically coupled oscillators can be proven following [PS07]. The dynamics of two Andronov-Hopf oscillators with symmetric diffusive linear coupling is given by:

$$\begin{aligned} \dot{x}_1 &= (1 - (x_1^2 + y_1^2))x_1 - \omega y_1 + k(x_2 - x_1) \\ \dot{y}_1 &= (1 - (x_1^2 + y_1^2))y_1 + \omega x_1 + k(y_2 - y_1) \\ \dot{x}_2 &= (1 - (x_2^2 + y_2^2))x_2 - \omega y_2 + k(x_1 - x_2) \\ \dot{y}_2 &= (1 - (x_2^2 + y_2^2))y_2 + \omega x_2 + k(y_1 - y_2) \end{aligned}$$

with $k > 0$. In vector notation (using $\mathbf{x}_i = [x_i, y_i]^T$, $i = 1, 2$, and the definition according to equation 2.17) one obtains:

$$\begin{aligned} \begin{bmatrix} \dot{\mathbf{x}}_1 \\ \dot{\mathbf{x}}_2 \end{bmatrix} &= \begin{bmatrix} \mathbf{f}(\mathbf{x}_1) \\ \mathbf{f}(\mathbf{x}_2) \end{bmatrix} - k \underbrace{\begin{bmatrix} \mathbf{I} & -\mathbf{I} \\ -\mathbf{I} & \mathbf{I} \end{bmatrix}}_{\mathbf{L}_{(2)}} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} \\ \Leftrightarrow \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}) - k \mathbf{L}_{(2)} \mathbf{x} \end{aligned} \tag{2.19}$$

where $\mathbf{x} = [\mathbf{x}_1^T, \dots, \mathbf{x}_N^T]^T$. According to the definition discussed in Section 2.3.3, a flow-invariant manifold \mathcal{M} of this system is given by the linear 2-dimensional subspace that is defined by the linear relationship $\mathbf{x}_1 = \mathbf{x}_2$. For points on this

manifold the coupling term vanishes, and the solution of the coupled system coincides with the solutions of the uncoupled individual oscillators. By interchanging the columns of the matrix $\mathbf{L}_{(2)} - \lambda\mathbf{I}$ it is easy to show that

$$\det(\mathbf{L}_2 - \lambda\mathbf{I}) = \begin{vmatrix} (1-\lambda)\mathbf{I} & -\mathbf{I} \\ -\mathbf{I} & (1-\lambda)\mathbf{I} \end{vmatrix} = \begin{vmatrix} 1-\lambda & 0 & -1 & 0 \\ 0 & 1-\lambda & 0 & -1 \\ -1 & 0 & 1-\lambda & 0 \\ 0 & -1 & 0 & 1-\lambda \end{vmatrix}$$

Interchanging 2nd and 3rd columns and rows, we get:

$$= \begin{vmatrix} 1-\lambda & -1 & 0 & 0 \\ -1 & 1-\lambda & 0 & 0 \\ 0 & 0 & 1-\lambda & -1 \\ 0 & 0 & -1 & 1-\lambda \end{vmatrix} = \begin{vmatrix} 1-\lambda & -1 \\ -1 & 1-\lambda \end{vmatrix}^2 = \lambda^2(\lambda-2)^2$$

This implies that the matrix $\mathbf{L}_{(2)}$ has rank 2. Its nullspace is 2-dimensional and thus coincides with \mathcal{M} . If according to Section 2.3.3, the matrix \mathbf{V} is a projector onto \mathcal{M}^\perp this implies that the matrix $\mathbf{V}\mathbf{L}_{(2)}\mathbf{V}^T$ has only the eigenvalues 2.

The Jacobian for a single oscillator is given by

$$\mathbf{J}(\mathbf{x}_i) = \frac{\partial \mathbf{f}}{\partial \mathbf{x}_i} = \begin{bmatrix} -(x_i^2 + y_i^2 - 1) - 2x_i^2 & -2x_i y_i - \omega \\ -2x_i y_i + \omega & -(x_i^2 + y_i^2 - 1) - 2y_i^2 \end{bmatrix}$$

implying

$$|\mathbf{J}_s(\mathbf{x}_i) - \lambda\mathbf{I}| = (1 - r^2 - \lambda)(1 - 3r^2 - \lambda)$$

with $r^2 = x_i^2 + y_i^2$. The eigenvalues of the matrix $\mathbf{J}_s(\mathbf{x}_i)$ are thus bounded by 1 from above.

Using the derived bounds for the eigenvalues, a sufficient condition for global exponential convergence of the coupled oscillator system can be derived from equation 2.15:

$$\lambda_{\min}(\mathbf{V}(k\mathbf{L}_{(2)})\mathbf{V}^T) = 2k > \sup_{\mathbf{x}, t} \lambda_{\max} \left(\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right)_s \right) = 1 \quad (2.20)$$

This implies that a sufficiently strong coupling with $k > 1/2$ guarantees the global exponential convergence against a stable behavior. Hence, the two oscillators synchronize for non-zero initial conditions, which is shown in an example (Figure 2.10), where the initial conditions of two oscillators differ from each other and become synchronized using bidirectional couplings.

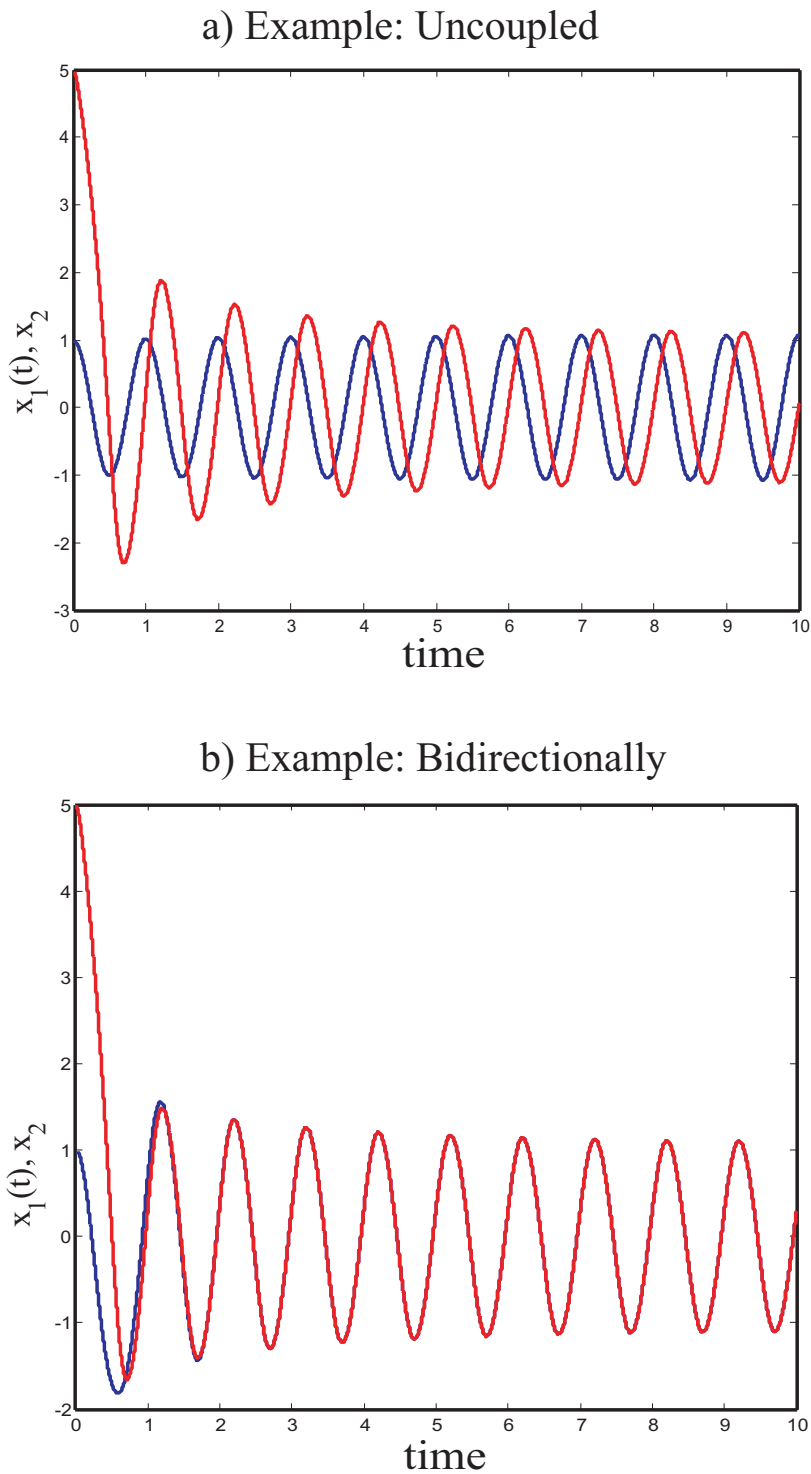


Figure 2.10: a) Two uncoupled oscillators show no synchronization behavior; b) Two bidirectionally coupled oscillators synchronize.

2.4.1.2 Symmetric Coupling of Three Oscillators

The same proof can be extended for the system of three coupled Hopf oscillators, where we use the diffusive coupling term with $k > 0$:

$$\begin{aligned}
 \dot{x}_1 &= (1 - (x_1^2 + y_1^2)) x_1 - \omega y_1 + k(x_2 - x_1) + k(x_3 - x_1) \\
 \dot{y}_1 &= (1 - (x_1^2 + y_1^2)) y_1 + \omega x_1 + k(y_2 - y_1) + k(y_3 - y_1) \\
 \dot{x}_2 &= (1 - (x_2^2 + y_2^2)) x_2 - \omega y_2 + k(x_1 - x_2) + k(x_3 - x_2) \\
 \dot{y}_2 &= (1 - (x_2^2 + y_2^2)) y_2 + \omega x_2 + k(y_1 - y_2) + k(y_3 - y_2) \\
 \dot{x}_3 &= (1 - (x_3^2 + y_3^2)) x_3 - \omega y_3 + k(x_1 - x_3) + k(x_2 - x_3) \\
 \dot{y}_3 &= (1 - (x_3^2 + y_3^2)) y_3 + \omega x_3 + k(y_1 - y_3) + k(y_2 - y_3)
 \end{aligned}$$

Which can be described by the equations:

$$\begin{bmatrix} \dot{\mathbf{x}}_1 \\ \dot{\mathbf{x}}_2 \\ \dot{\mathbf{x}}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{f}(\mathbf{x}_1) \\ \mathbf{f}(\mathbf{x}_2) \\ \mathbf{f}(\mathbf{x}_3) \end{bmatrix} - k \underbrace{\begin{bmatrix} 2\mathbf{I} & -\mathbf{I} & -\mathbf{I} \\ -\mathbf{I} & 2\mathbf{I} & -\mathbf{I} \\ -\mathbf{I} & -\mathbf{I} & 2\mathbf{I} \end{bmatrix}}_{\mathbf{L}_{(3)}} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \end{bmatrix} \quad (2.21)$$

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) - k\mathbf{L}_3\mathbf{x} \quad (2.22)$$

where $\mathbf{x} = [\mathbf{x}_1^T, \dots, \mathbf{x}_N^T]^T$. In this case, the characteristic equation for $\mathbf{L}_{(3)}$ is: $\det(\mathbf{L}_{(3)} - \lambda\mathbf{I}) = \lambda^2(\lambda - 3)^4 = 0$, where all non-zero eigenvalues of $\mathbf{L}_{(3)}$ are equal to 3 and it follows by the same argumentation as in Section 2.4.1.1:

$$\lambda_{\min}(\mathbf{V}(k\mathbf{L}_{(3)})\mathbf{V}^T) = 3k > \sup_{\mathbf{x}, t} \lambda_{\max} \left(\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right)_s \right) = 1 \quad (2.23)$$

Hence, $k > 1/3$ presents the derived condition for the global exponential convergence to the stable phase-locking behavior of three coupled Hopf oscillators.

Due to the construction of the coupling, the flow-invariant manifold is two-dimensional. Here the flow-invariant manifold is spanned with the set of all trajectories when all oscillators behave identically, and one Hopf oscillator has 2 degrees of freedom. But $\dim(\text{Null}(\mathbf{L})) = 2$ holds also in this case, so that $\text{Null}(\mathbf{L})$ and the flow-invariant manifold coincide.

2.4.1.3 Symmetric All-to-All Coupling of N Oscillators

The last analysis can be extended to any number N of coupled oscillators (Figure 2.11). In this case, the $2N$ -dimensional square matrix \mathbf{L} has the form:

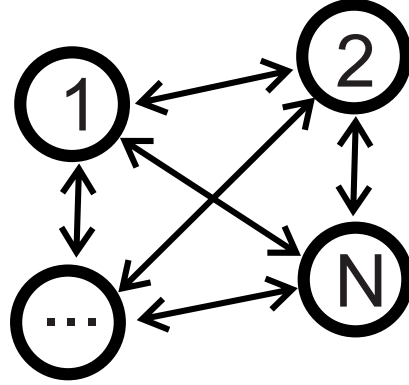


Figure 2.11: All-to-all coupling of N oscillators, with equal symmetrical coupling gains k .

$$\mathbf{L} = \begin{bmatrix} (N-1) & 0 & -1 & 0 & \dots \\ 0 & (N-1) & 0 & -1 & \dots \\ -1 & 0 & (N-1) & 0 & \dots \\ 0 & -1 & 0 & (N-1) & \dots \\ \dots & \dots & \dots & \dots & \dots \end{bmatrix}$$

i.e., $L_{ii} = N - 1$ and $L_{ij} = -1$ if $i \neq j$ and $(i + j) \bmod 2 = 0$, and $L_{ij} = 0$ otherwise. By rearranging the columns and rows, this matrix can be restructured in the form:

$$\mathbf{L} = \begin{bmatrix} \mathbf{L}_G & 0 \\ 0 & \mathbf{L}_G \end{bmatrix} \quad (2.24)$$

where:

$$\mathbf{L}_G = \begin{bmatrix} (N-1) & -1 & -1 & \dots \\ -1 & (N-1) & -1 & \dots \\ -1 & -1 & (N-1) & \dots \\ \dots & \dots & \dots & \dots \end{bmatrix} \quad (2.25)$$

Note that $\mathbf{L}_G = N\mathbf{I} - \mathbf{1}\mathbf{1}^T$. The matrix $\mathbf{1}\mathbf{1}^T$ has rank 1 and the eigenvector $\mathbf{1}$ with the eigenvalue N , while all other eigenvalues are 0. From $\det(\mathbf{L}_G - \lambda\mathbf{I}) = \det(-\mathbf{1}\mathbf{1}^T - (\lambda - N)\mathbf{I}) = 0$ follows that the matrix \mathbf{L}_G has one eigenvalue 0 and all other $N - 1$ eigenvalues are N . Resulting from this and equation 2.24 the two eigenvalues of the matrix \mathbf{L} are 0, while all non-zero eigenvalues are N . As for equation 2.20 one obtains the inequality

$$Nk > \sup_{\mathbf{x}, t} \lambda_{\max} \left(\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right)_s \right) = 1$$

Global exponential convergence to a stable synchronized solution is thus guaranteed for $k > 1/N$.

2.4.2 Symmetric Couplings: More General Structure

This subsection deals with systems with more general symmetric couplings of N number of oscillator, following the proceeding in [WS05]. Therefore, suppose equal coupling gains k . The corresponding dynamics is then:

$$\dot{\mathbf{x}}_i = \mathbf{f}(\mathbf{x}_i) + k \sum_{j \in \mathcal{N}_i} (\mathbf{x}_j - \mathbf{x}_i), \quad \forall i = 1, \dots, N \quad (2.26)$$

where \mathcal{N}_i denotes the set of indices of all oscillators that are coupled with oscillator i . The couplings are assumed to be bidirectional, defining an undirected graph of couplings. This implies $j \in \mathcal{N}_i$ iff $i \in \mathcal{N}_j$. By construction the coupling graph is *balanced*, i.e. the sum of the (weighted) connections towards each oscillator equals the sum of (weighted) connections away from this oscillator. The corresponding symmetric Laplacian matrix \mathbf{L} has a block structure, where the blocks at positions (i, i) are given by $-\mathbf{I}$ where the i -th diagonal block is given by $n_i \mathbf{I}$, n_i signifying the number of elements in \mathcal{N}_i :

$$\mathbf{L} = \begin{bmatrix} \ddots & \vdots & & \vdots & & \\ \dots & \mathbf{I} & \dots & -\mathbf{I} & \dots & \\ & \vdots & \ddots & \vdots & & \\ \dots & -\mathbf{I} & \dots & \mathbf{I} & \dots & \\ & \vdots & & \vdots & \ddots & \end{bmatrix}_{N \times N} \quad (2.27)$$

Like in the previous sections, this matrix, by appropriate sorting of columns and rows, can be brought in the form $\begin{bmatrix} \mathbf{L}_G & 0 \\ 0 & \mathbf{L}_G \end{bmatrix}$, where \mathbf{L}_G is called Laplacian matrix of the coupling graph.

Since the network is balanced, the sum of the rows of this matrix are zero. This implies that $\mathbf{1}$ is an eigenvector with eigenvalue 0.

Again, the block structure implies that all eigenvalues of \mathbf{L}_G appear twofold in the matrix \mathbf{L} . Consequently, two of its eigenvalues are zero, independent of the form of the sets \mathcal{N}_i .

Following the argumentation in the last sections one can derive a necessary con-

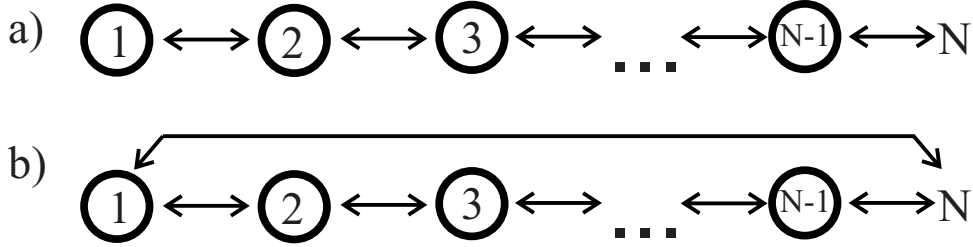


Figure 2.12: Symmetric coupling. a) Chain and b) ring coupling of N oscillators.

dition for the exponential convergence from equation 2.15:

$$\lambda_{\min}(\mathbf{V}(k\mathbf{L})_s\mathbf{V}^T) = k\lambda_{\mathbf{L}}^+ > \sup_{\mathbf{x},t} \lambda_{\max}\left(\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)_s\right) = 1 \quad (2.28)$$

Here, $\lambda_{\mathbf{L}}^+$ stands for the smallest non-zero eigenvalue of the matrix \mathbf{L}_G that depends on the form of the coupling. The matrix \mathbf{V} defines the projection to the orthogonal complement of the flow-invariant manifold $\mathbf{x}_1 = \dots = \mathbf{x}_n$. The condition for exponential convergence thus is

$$k > 1/\lambda_{\mathbf{L}}^+ \quad (2.29)$$

Considering symmetric coupling, different coupling structures can be analyzed in this way to obtain synchronization conditions. Beside diffusive coupling, which was discussed earlier in Section 2.4.1.3, Figure 2.12 shows additionally coupling structures: A symmetric chain structure of a set of N oscillators (a) and a two-way-ring structure on the right side (b). In the case of the two-way-chain structure, the first nonzero eigenvalue of the matrix \mathbf{L}_G can be shown to be

$$\lambda_{\mathbf{L}}^+ = 2(1 - \cos(\pi/N))$$

Whereas, for a symmetric-coupled ring structure, the synchronization condition is $1/\lambda_{\mathbf{L}}^+$ [WS05], where

$$\lambda_{\mathbf{L}}^+ = 2(1 - \cos(2\pi/N)) \quad (2.30)$$

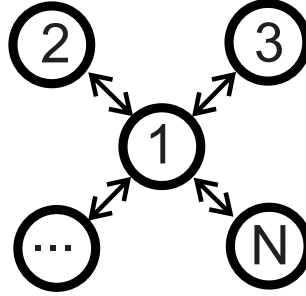


Figure 2.13: Star coupling

Finally, a *star coupling* of $N > 2$ oscillators can be interpreted as a network, where $N - 1$ oscillators are not connected with each-other, but only with the central node of the star. Moreover, the couplings are again bidirectionally and the coupling gains for all connections are identical. The corresponding coupling graph is illustrated in Figure 2.13. If the first oscillator represents the center-node of the star structure, it implies for the elements of the Laplacian matrix, that $(\mathbf{L}_G)_{1,1} = N - 1$, $(\mathbf{L}_G)_{i,i} = 1$, $(\mathbf{L}_G)_{1,i} = -1$ for $\forall i > 1$, while all other entries are zero $(\mathbf{L}_G)_{i,j} = 0$.

It can be shown that the eigenvalues of this matrix are then 0, 1 ($(N - 2)$ times), and N . Hence, $\lambda_{\mathbf{L}}^+ = 1$ it follows the stability condition

$$k > 1 \quad (2.31)$$

in order to achieve a synchronization of the network.

2.4.3 Leader-Group Coupling

Assume a scenario, where a single oscillator is coupled to group of N identical oscillators that are already synchronized. The underlying dynamics is defined by:

$$\dot{\mathbf{x}}_0 = \mathbf{f}(\mathbf{x}_0) - k \left(N\mathbf{x}_0 - \sum_i \mathbf{x}_i \right) = \mathbf{f}(\mathbf{x}_0) - kN(\mathbf{x}_0 - \mathbf{x}_1)$$

A particular solution of this system is $\mathbf{x}_0 = \mathbf{x}_1$. If the system is partially contracting in \mathbf{x}_0 this implies the exponential convergence of the follower state \mathbf{x}_0 against the equilibrium state \mathbf{x}_1 of the other oscillators. This condition is obviously fulfilled if

$$kN > \sup_{\mathbf{x}, t} \lambda_{\max} \left(\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right)_s \right) = 1 \quad (2.32)$$

In a leader scenario, the single oscillator feeds unidirectionally into all other N oscillators with the same coupling strength α , but not vice versa, as it is illustrated in Figure 2.14. This situation is described by the dynamics (for $1 \leq i \leq N$):

$$\begin{cases} \dot{\mathbf{x}}_0 = \mathbf{f}(\mathbf{x}_0) \\ \dot{\mathbf{x}}_i = \mathbf{f}(\mathbf{x}_i) + k\mathbf{I} \sum_{j \in \mathcal{N}_i} (\mathbf{x}_j - \mathbf{x}_i) + \alpha(\mathbf{x}_0 - \mathbf{x}_i) \end{cases}$$

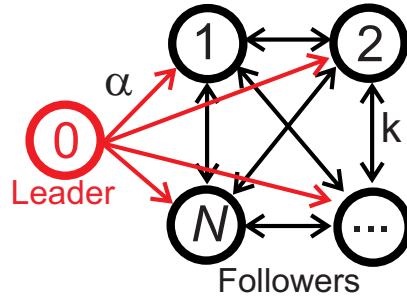


Figure 2.14: Leader-group coupling: Single oscillator feeds unidirectionally into all other N oscillators with the same coupling strength k .

Since the leader oscillator does not receive external inputs, it oscillates autonomously, and \mathbf{x}_0 can be treated as external input. Applying partial contraction analysis to the second equation, one obtains the dynamics by using the same definitions as in Section 2.3.3, where $\tilde{\mathbf{x}}_0 = [\mathbf{x}_0^T, \dots, \mathbf{x}_0^T]^T$ and $\mathbf{x} = [\mathbf{x}_1^T, \dots, \mathbf{x}_N^T]^T$.

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) - k\mathbf{L}\mathbf{x} - \alpha\mathbf{x} + \alpha\tilde{\mathbf{x}}_0 \quad (2.33)$$

This implies

$$\mathbf{J}(\mathbf{x}, t) = \mathbf{D}(\mathbf{x}, t) - k\mathbf{L} - \alpha\mathbf{I}$$

and the contraction condition:

$$\lambda_{\min}(k\mathbf{L}_G + \alpha\mathbf{I}) > \sup_{\mathbf{x}, t} \lambda_{\max} \left(\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right)_s \right) = 1 \quad (2.34)$$

For the special case that the N oscillators (except for the leader oscillator) are symmetrically coupled all-to-all, the contraction condition becomes

$$kN + \alpha > 1 \quad (2.35)$$

This implies that for $kN < 1$, a contracting behavior can still be guaranteed when the coupling α to the leader oscillator is sufficiently strong. The minimum convergence rate is then given by $\rho_c = kN + \alpha$. Figure 2.15 presents a simulation of

two unidirectionally coupled oscillators, which start in anti-synchronization and become coordinated.

Different coupling structures and coupling strengths have a diverse impact on the behavior of the oscillator networks. By understanding the stability properties of such systems using partial contraction theory, one is able to control such complex interactions between an arbitrary number of nonlinear-dynamic primitives to obtain a stable and coordinated behavior.

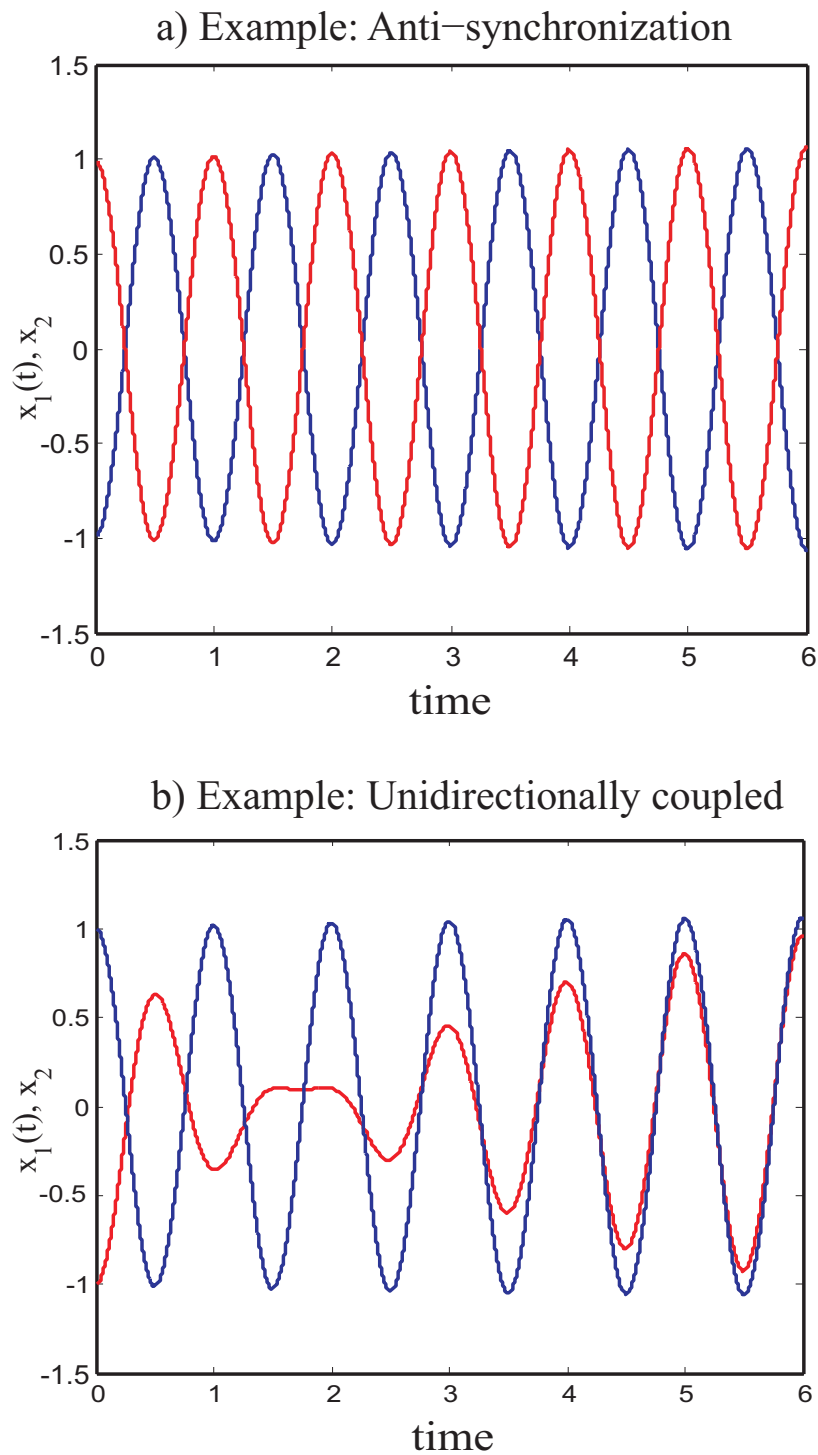


Figure 2.15: a) Two oscillators with opposite initial conditions $(1, -1)$ are uncoupled resulting in an anti-synchronization behavior. b) Two unidirectionally coupled oscillators synchronize.

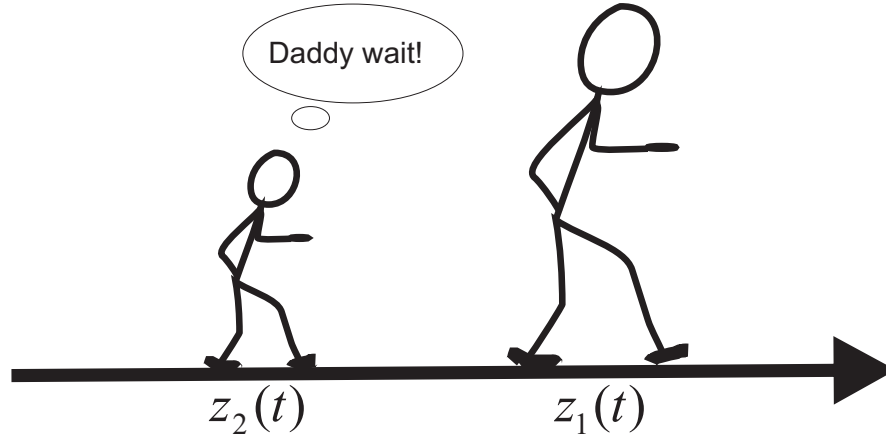


Figure 2.16: Following the Leader: $z_1(t)$ and z_2 define the position of the follower and the leader at time t .

2.5 Distance-Frequency Control

The following section deals with the stability analysis of dynamics of simple following behavior. As illustrated in Figure 2.16, $z_1(t)$ and $z_2(t)$ specify the positions of the leader and the follower. The follower is modeled by a Andronov-Hopf oscillator, whereas the leader is just represented by his position $z_2(t)$. Let the distance-frequency coupling for the follower be given by:

$$\omega(t) = \omega_0 + m(z_1(t) - z_2(t)) \quad (2.36)$$

where m denotes some positive constant ($m > 0$). Furthermore, we assume that ω_0 is the frequency of the follower, which corresponds to the propagation speed of the leader (ω_0 can be estimated by observing the leader).

The last equation specifies a position control, where (for constant propagation speed of the first agent) $\omega(t)$ changes until the position of the second avatar matches the one of the first, i.e. $z_1(t) \approx z_2(t)$ for large t .

In fact, the propagation speed depends in a very complex way on the phase $\phi(t)$ of the oscillator through the highly nonlinear kinematics. Nevertheless, as a first example we analyze the extremely simplified case where the relationship between propagation speed and phase of the oscillator can be approximated by the linear relationship

$$\dot{z}_2(t) = P\omega(t) = P\dot{\phi}(t) \quad (2.37)$$

where P is some positive constant.

2.5.1 Analysis

Below the following simplified notations are used: $z_1(t) = A(t)$ for the position of the leader z_1 at time t and $z(t) = z_2(t) = P \int \omega(t) dt$ for the position of the follower z_2 at time t .

2.5.1.1 Linear Approximation of Kinematics

From the previous assumption we derive

$$\begin{aligned} \dot{z}(t) &= P\omega(t) \\ \omega(t) &= \omega_0 + m(A(t) - z(t)) \end{aligned} \quad (2.38)$$

Insertion of the last equation in the first yields:

$$\dot{z}(t) = P(\omega_0 + m(A(t) - z(t))) \quad (2.39)$$

Differentiation with respect to time results in:

$$\ddot{z}(t) = P\dot{\omega}(t) = -P^2m\omega(t) + Rm\dot{A}(t) \quad (2.40)$$

$$\Rightarrow \dot{\omega}(t) = m(\dot{A}(t) - P\omega(t)) \quad (2.41)$$

This is a linear dynamics for $\omega(t)$ that is obviously contracting for $mP > 0$. If the propagation speed of the leader is constant, (which means, by definition of ω_0 that $\dot{A}(t) = P\omega_0$), then $\omega(t) \rightarrow \omega_0$, $\dot{z}(t) \rightarrow P\omega_0$, and from equation 2.18 follows that $z(t) \rightarrow A(t)$. More abstractly this implies that coupled Hopf oscillators of this type with the specified linear approximation of the propagation model specify a cascade system, where the eigenfrequency as an external parameter influences the oscillator(s), but the state of the oscillator does not feed back in the dynamics of $\omega(t)$.

2.5.1.2 Nonlinear Approximation of Kinematics

A more accurate description of the kinematics driven by general dynamical system results in a dependency of the propagation speed of the variables $\phi(t)$, $\dot{\phi}(t)$, $r(t)$, $\dot{r}(t)$. Due to the complexity of this nonlinear relationship, we believe that this problem cannot be solved exactly. Instead a bound has been computed by deriving appropriate constraints for the relationship:

$$\dot{z}(t) = g(\phi(t), r(t), \dot{\phi}(t), \dot{r}(t)) \quad (2.42)$$

The nonlinear function $g(\cdot)$ of forward kinematics has the effect that in this case the oscillator state feeds back into the $\omega(t)$ dynamics, which makes the analysis

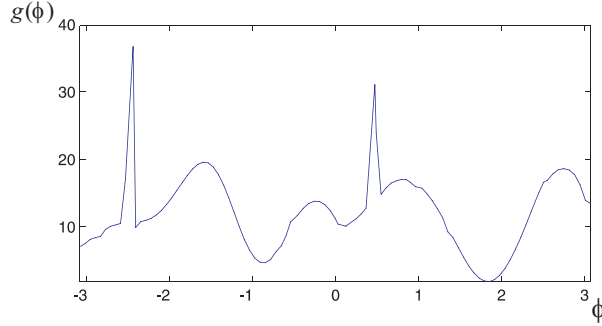


Figure 2.17: Propagation velocity dependent on the gait cycle phase ϕ .

of the system much harder. We propose to proceed in the following ways:
The first extension is to assume

$$\dot{z}(t) = \dot{\phi}(t)g(\phi(t)) = \omega(t)g(\phi(t)) \quad (2.43)$$

where $\omega(t)g(\phi(t))$ denotes the propagation speed for the avatar driven by a single Hopf oscillator resting on its limit cycle. An example for an empirically determined function $g(\phi(t))$ is shown in Figure 2.17, where the propagation velocity $g(\phi(t))$ was estimated for a constant $\omega = 1$. In this case, the decoupling of the $\omega(t)$ dynamics and the oscillator dynamics remains valid, and the system dynamics for one avatar driven by single Hopf oscillator on its limit cycle is given by two differential equations:

$$\begin{cases} \dot{z}(t) = \dot{\phi}(t)g(\phi(t)) = \dot{G}(\phi(t)) \\ \dot{\phi}(t) = \omega_0 + m(A(t) - z(t)) \end{cases} \quad (2.44)$$

where $G(\phi) = \int_0^\phi g(\phi)d\phi + const$. The position of the follower is $z(t)$, and $A(t)$ is the position of the leader avatar. Here m is a positive coupling constant and $g(\phi)$ is positive function. $dG(\phi)/d\phi = g(\phi)$ and $G(\phi)$ is a strictly increasing function, for which we can choose initial conditions so that $G(0) = 0$. Using the identity $z(\phi(t)) = G(\phi(t)) - c$, the system can be rewritten as

$$\dot{\phi}(t) = \omega_0 + m(A(t) - G(\phi(t)) + c) \quad (2.45)$$

where c is the constant that depends on the initial phase and position of the follower. The symmetrized Jacobian of the system is

$$\mathbf{J}_s = -mg(\phi(t))$$

which is always negative for $m > 0$ and guarantees that the system is contracting. This means that the trajectories of all following avatars (starting with the same

initial relations to the leader as determined by c) converge to the same single trajectory.

If the dynamics for the position control of the follower (equation 2.36) is extended by including a nonlinear coupling ξ , i.e. sigmoid function, used for the ‘father-son example’ in Section 5.1.1.2, contraction is guaranteed when the nonlinear coupling is positive with $\xi > 0$. Therefore, the distance-frequency coupling of the follower is denoted by:

$$\omega(t) = \omega_0 + \xi(z_1(t) - z_2(t)) \quad (2.46)$$

Corresponding to equation 2.45 the dynamics can be written in

$$\dot{\phi}(t) = \omega_0 + \xi(A(t) - G(\phi(t)) + c)$$

and it follows that the symmetrized Jacobian is

$$\mathbf{J}_s = -\dot{\xi}(A(t) - G(\phi(t)) + c)g(\phi(t))$$

where for $\dot{\xi} > 0$ a contracting behavior is guaranteed.

In this section, contraction theory as a tool for stability analysis of complex nonlinear systems was presented and applied to exploit this approach for obtaining stability properties from coupled nonlinear limit-cycle oscillators. Indeed, systems of different coupling structures with arbitrary size were investigated; Additionally simple leader-follower dynamics have been discussed. The computed stability properties can be used for the controlled application in character animation.

CHAPTER 3

Blind Source Separation

The aim of dimensional reduction is to decrease redundancy in large data set of parameters or features, which can be summarized into a set of fewer dimensions. In this way, the extracted information reformulates the original data using less parameters [LV07]. Dimension reduction of information is currently an important topic in many different domains such as in biology, engineering, and other areas of applications dealing with data processing.

Especially in entertainment software, like video games with limited memory, it is of high interest to reduce the dimension of the original data without losing quality and make the data accessible for the environment (applicants). For instance, complex body movements represent highly redundant trajectory data and leads to a tedious process for animating virtual humans due to the ‘degrees of freedom problem’ (Section 1.1). Therefore, a more compact representation of the joint trajectories is desired to simplify the treatment of the many degrees of freedom by separating the movement into simpler components [IPL04, FH05]. For this purpose, unsupervised learning techniques, such as blind source separation, can be applied and serves as the main topic of this chapter.

3.1 General Approach to Blind Source Separation



Figure 3.1: The *cocktail-party problem* presents the classical BSS problem since the number of speakers is larger than the observed mixtures recorded by the two microphones x_1 and x_2 . Hence, this example represents an under-determined problem because several simultaneously active signal sources can be separated at different spatial locations by assuming mutual independence of the sources.

Blind Source Separation (BSS) can be applied to a variety of situations, as it is in the case of speech processing [BS95, OPR05]. Although no information regarding the source signals or the mixing process is given, source signals from a set of mixed signals can be recovered. In fact, BSS relies on the assumption that the source signals are mutually independent and, by superpositioning them, the mixed signals can be reconstructed. Linear BSS problems can be summarized into the following three kinds of mixtures:

1. Instantaneous mixtures
2. Anechoic Mixtures
3. Convolutional Mixtures

The classical example for the source separation problem is been presented in the well-known 'cocktail party problem' [HO01, PLKP07] and is illustrated in Figure 3.1. Hereby, the task of the BSS algorithm is to extract the voice of a single person from an ensemble or mixtures of different voices corrupted by music and background noises.

In many cases the number of mixtures (m microphones) are larger than the number of sources (n speakers) so that $m > n$ and it is defined as an **over-determined** mixing problem. To solve such a problem, the application of dimensionality reduction methods is usually the first step to follow.

Vice versa, if the number of sources is smaller than the number of mixtures ($m < n$), the model is known as an **under-determined** mixing model.

However, the cocktail party problem can be mathematically formalized as follows:

$$x_i(t) = \sum_{j=1}^n w_{ij} \cdot s_j(t) \quad i = 1, \dots, m. \quad (3.1)$$

where x_i denotes the recorded mixtures of the original source signals (speakers) s_j in time t and w_{ij} defines the mixing weights. This simplest form of the BSS problem is referred to as **instantaneous mixtures** (Section 3.2).

Anechoic mixtures (Section 3.3) can be seen as an extension of the instantaneous mixing model with time delays. In regard to the 'cocktail party problem', the party is located in a reverberant environment, where the source signals are recorded with different time delays due to reflections on the walls. Considering the different times of arrival –delays (τ)– of the sounds, the mixing model is termed by

$$x_i(t) = \sum_{j=1}^n w_{ij} \cdot s_j(t - \tau_{ij}) \quad i = 1, \dots, m. \quad (3.2)$$

and can be seen as a special case of a positive convolutive model (see [Oml10]).

3.1.1 BSS For Motion Analysis

As it was discussed earlier in Section 1.3.2 a classical assumption in motor control implies that complex behavior can be reconstructed by an appropriate linear combination of simpler movement primitives or synergies. For extracting the basic components from motion data, instantaneous BSS algorithms like principal component analysis (PCA), independent component analysis (ICA), or non-negative matrix factorization (NMF) have been successfully applied [SFS98, IPL04].

Corresponding to the instantaneous model in equation 3.1, $x_i(t)$ represents the

recorded motion data and is modeled as a linear combination of weighted source signals $s_j(t)$.

However, to separate movement components, the application of instantaneous BSS algorithms seem to be a common approach and has been also applied for full-body motion capture trajectories.

ICA has been used on individual joint angles trajectories; a comparison between the components among joints show a similarity in shape but a disparity of phase (Figure 3.2).

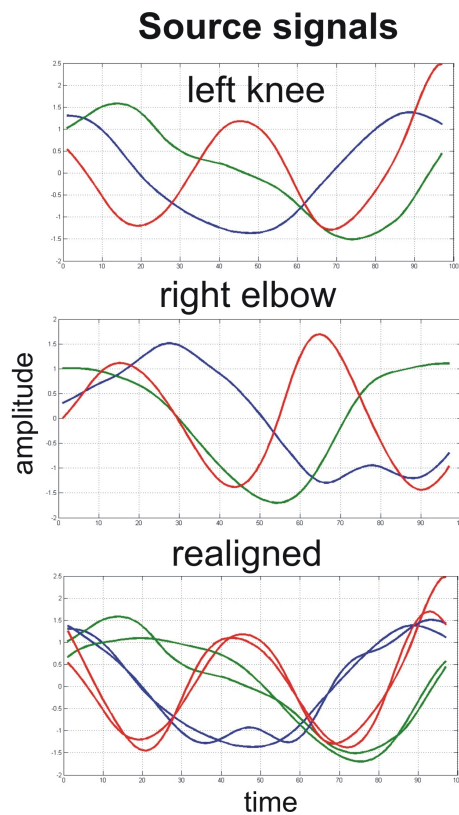
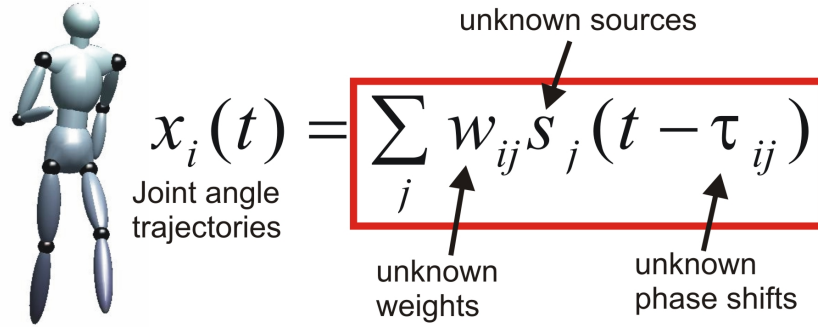


Figure 3.2: Extraction of primitives by using ICA on individual joint angles. Source signals of individual joints have a similar shape, but are time-shifted against each other.

From this observation a mixture model with time delays τ_{ij} among sources might be more convenient and corresponds to the anechoic mixing model 3.2. More details can be found in [Oml10].



$$x_i(t) = \sum_j w_{ij} s_j(t - \tau_{ij})$$

Joint angle trajectories

unknown sources

unknown weights

unknown phase shifts

Figure 3.3: Generative mixing model: Superposition of independent shift-invariant source signals.

3.2 Instantaneous Blind Source Separation

In order to find a suitable representation of the data, an appropriate transformation is needed and states a common problem in many scientific areas, as in statistics or signal processing [Hyv99, LV07]. In mathematical terms the task can be reformalized as in the following equation. A function f is sought, which expresses an m -dimensional random variable \mathbf{x} as an n -dimensional transform $\mathbf{s} = (s_1, s_2, \dots, s_n)^T$ and is defined by

$$\mathbf{s} = f(\mathbf{x})$$

In general cases, one seeks for a linear transformation of the random variable, so that

$$\mathbf{s} = \mathbf{B}\mathbf{x}$$

where \mathbf{B} defines the matrix, which has to be determined. Principal component analysis, independent component analysis, and non-negative matrix factorization are few of many methods for such a linear representation.

3.2.1 Principal Components Analysis

The **Principal Components Analysis** (PCA), also known as (discrete) Karhunen-Loève transform, or the Hotelling transform, represents a classical example for the linear transformation of data. Moreover, it is a common method for dimension reduction, and feature extraction [Bis07, Hyv99]. In fact, PCA belongs to the second-order method and uses the information from the covariance matrix of the data vector \mathbf{x} . Additionally to the assumption on linearity, the assumption of normal or Gaussian distribution of the random variable is required.

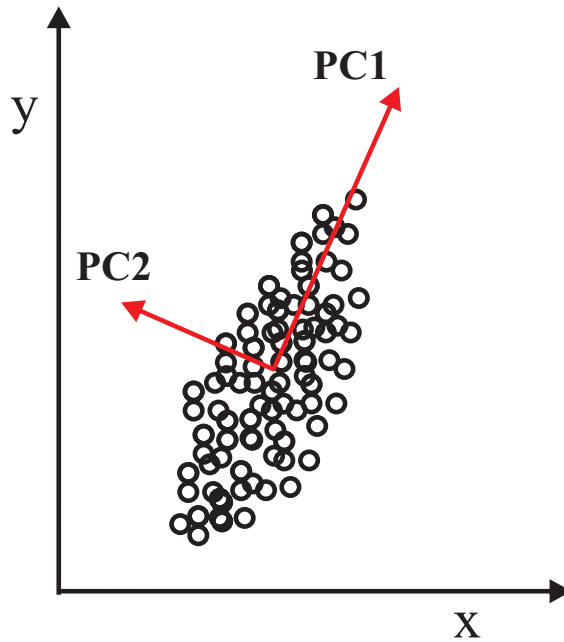


Figure 3.4: PCA projects the data along the directions where the data varies the most. These directions are determined by the eigenvectors of the covariance matrix corresponding to the largest eigenvalues. The magnitude of the eigenvalues corresponds to the variance of the data along the eigenvector directions.

Before applying PCA on the variable \mathbf{x} , the mean is often subtracted in order to center the data, which can be viewed as a first transformation:

$$\mathbf{x} = \mathbf{x}_0 - E\{\mathbf{x}_0\}$$

where \mathbf{x}_0 denotes the original non-centered variable. The goal of PCA is to find the principal components b_1, b_2, \dots, b_n and their weights s_1, s_2, \dots, s_n , which explain the maximum amount of variance by n linearly transformed components. For this purpose, the direction of the first principal component \mathbf{b}_1 is computed by

$$\mathbf{b}_1 = \arg \max_{\|\mathbf{b}\|=1} E\{(\mathbf{b}^T \mathbf{x})^2\}$$

and has the same dimension m as the random data vector \mathbf{x} , which explains the largest variance of the data (see Figure 3.4). Having the first principal components $k-1$, the k -th principal component is then determined as the principal component of the residual:

$$\mathbf{b}_k = \arg \max_{\|\mathbf{b}\|=1} E\{[\mathbf{b}^T (\mathbf{x} - \sum_{i=1}^{k-1} \mathbf{b}_i \mathbf{b}_i^T \mathbf{x})]^2\}$$

The sources weights are then given by $s_i = \mathbf{b}_i^T \mathbf{x}$. For the computation of \mathbf{b}_i the covariance matrix \mathbf{C} of the data \mathbf{x} is needed, where $\mathbf{C} = E\{\mathbf{x}\mathbf{x}^T\}$. From \mathbf{C} the eigenvalues $\lambda_1 > \lambda_2 > \dots > \lambda_m$ and then the eigenvectors of \mathbf{C} are computed. Since PCA emphasized the dimensionality reduction of the observed data, one usually chooses $n < m$, so that \mathbf{b}_i define the eigenvectors corresponding to the n largest eigenvalues of \mathbf{C} .

3.2.2 Independent Component Analysis

In comparison to PCA, the **independent component analysis** (ICA) belongs to a higher order method and use information on the distribution of \mathbf{x} that is not contained in the covariance matrix. Hence, to apply ICA successfully to the generative instantaneous mixing model (see also Section 3.1):

$$\mathbf{x} = \mathbf{A}\mathbf{s}$$

assumptions, besides the centering of \mathbf{x} and \mathbf{s} are required [CCPL05, VM01]:

- The latent variables s_i in the vector $\mathbf{s} = (s_1, \dots, s_n)^T$ are assumed to be independent.
- The independent components have non-Gaussian distribution.

The original sources \mathbf{s} can be recovered by multiplying the observed signals \mathbf{x} with the inverse of the mixing matrix $\mathbf{U} = \mathbf{A}^{-1}$, also known as the unmixing matrix. Thus, the estimation of the unmixing matrix \mathbf{U} is needed, to obtain the sought independent source signals \mathbf{y} :

$$\mathbf{y}(t) = \mathbf{U}\mathbf{x}(t)$$

Therefore, several approaches for the ICA problem have been developed [Hyv99, HO01]. The most common one is based on the algorithms for maximum likelihood or infomax estimation; and defines a (stochastic) gradient ascent of the objective function. For instance, the log-likelihood of the unmixing matrix \mathbf{U} is given by:

$$\log L(\mathbf{U}) = -\frac{1}{2} \log |\det \mathbf{U}\mathbf{U}^T| - \sum_{i=1}^n \log(p_i(y_i))$$

and can then be maximized by the natural gradient descent:

$$\Delta \mathbf{U} = [\mathbf{U}^T]^{-1} + E\{g(\mathbf{y})\mathbf{x}^T\}.$$

where $g(\cdot)$ denotes a nonlinear function, which must approximate the cumulative distribution function (CDF) of \mathbf{s} . In practice hyperbolic tangent and logistic functions are good estimates for $g(\cdot)$ [Oml2010].

3.2.3 Non-Negative Matrix Factorization

The Non-negative Matrix Factorization or NMF defines an algorithm, which was developed for 2-D images [LS01]. Given a non-negative data, NMF finds an approximate factorization of a matrix \mathbf{X} ($M \times N$) into factors \mathbf{A} ($M \times R$) and \mathbf{B} ($R \times N$) with the constraint that \mathbf{A} and \mathbf{B} must be non-negative.

$$\mathbf{X} \approx \mathbf{AB}$$

This equation can be re-written also as:

$$\mathbf{X} = \sum_{r=1}^R a_r b_r$$

where a_r denotes the vector to be the r th column of $\mathbf{A} = a_1, a_2, \dots, a_R$ and b_r to be the vector of the r th row of $\mathbf{B} = b_1, b_2, \dots, b_R^T$. Each vector a_r describe the basis feature in \mathbf{X} , whereas the corresponding b_r is the vector of coefficients of this feature. However, to find such a pair \mathbf{A} and \mathbf{B} that approximates the data matrix \mathbf{X} and which minimizes the error of reconstruction, two cost functions are defined [PAB⁺06, LS01].

The first of the two cost functions Υ is based on the Euclidean distance between \mathbf{X} and \mathbf{AB} :

$$\Upsilon(\mathbf{X}, \mathbf{AB}) = \|\mathbf{X} - \mathbf{AB}\|_F^2$$

This is lower bounded by zero, and vanishes if and only if $\mathbf{X} = \mathbf{AB}$. An alternative measure is the divergence (generalized version of *Kullback-Leibler* [OP06]) between \mathbf{X} and \mathbf{AB} :

$$\chi(\mathbf{X}||\mathbf{AB}) = \left\| \mathbf{X} \circ \log \frac{\mathbf{X}}{\mathbf{AB}} - \mathbf{X} + \mathbf{AB} \right\|$$

The denotation \circ is known as Hadamard or Schur product and defines an element-wise product, and division is also elementwise.

The NMF can now be written as an optimization problem:

$$\min_{\mathbf{A}, \mathbf{B}} D(\mathbf{X}||\mathbf{AB})$$

and $\mathbf{A}, \mathbf{B} \geq 0$. A gradient descent update rules of \mathbf{A} and \mathbf{B} can be applied to minimize the cost functions:

$$\mathbf{B} = \mathbf{B} \circ \frac{\mathbf{A}^T \cdot \mathbf{X}}{\mathbf{A}^T \cdot \mathbf{1}}, \quad \mathbf{A} = \mathbf{A} \circ \frac{\mathbf{X} \cdot \mathbf{B}^T}{\mathbf{1} \cdot \mathbf{B}^T}$$

where $\mathbf{1}$ is an $M \times N$ matrix with all its elements equal to unity. More details about the minimization of the cost function and the proof can be found in [WP05, LS01].

3.3 Anechoic Demixing using Wigner-Ville Transform

In Section 3.1 anechoic mixing model were already introduced as an example for modeling the human motion. Opposed to common blind source separation techniques, such as PCA or ICA, this mixing model allows for time shifts τ_{ij} of the sources in the linear mixture.

$$x_i(t) = \sum_{j=1}^n w_{ij} \cdot s_j(t - \tau_{ij}) \quad i = 1, \dots, m. \quad (3.3)$$

The functions s_j denote hidden source signals and the parameters w_{ij} are the mixing weights. The standard anechoic demixing techniques in acoustics treat under-determined problems, which is not appropriate for dimension reduction purposes as it is necessary for motion analysis.

Therefore, we apply a suitable new approach *over-determined problems*, for which the signals outnumber the sources. Moreover, the anechoic mixing model is independent of the dimension of the data and the relation between the number of sources and the number of data points [OG07].

In order to solve this anechoic mixture model, the time shifts (delays), source signals and mixing weights are determined by a blind source separation algorithm that is based on a time-frequency integral transform, which is called *Wigner-Ville spectrum* (WVS). In this way, the signals can be represented in the time-frequency domain, while assuming that $x(t)$ is a random process and $x^* = \bar{x}(-t)$ defines the reversed conjugated process:

$$V_x(t, f) = \int_{\tau} E \left\{ x\left(t + \frac{\tau}{2}\right) x^*\left(t - \frac{\tau}{2}\right) \right\} e^{-2\pi i \tau f} d\tau$$

Applying this integral transform to the anechoic mixing model 3.3 rewrites the following time-frequency domain:

$$\begin{aligned} V_x(t, \omega) &:= \int E \left\{ \sum_{j,k=1}^n w_{ij} \bar{w}_{ik} s_j\left(t + \frac{\tau}{2} - \tau_{ik}\right) s_k^*\left(t + \frac{\tau}{2} - \tau_{ik}\right) \right\} e^{-2\pi i \omega \tau} d\tau = \\ &\sum_{j,k=1}^n w_{ij} \bar{w}_{ik} \int E \left\{ s_j\left(t + \frac{\tau}{2} - \tau_{ik}\right) s_k^*\left(t + \frac{\tau}{2} - \tau_{ik}\right) \right\} e^{-2\pi i \omega \tau} d\tau \end{aligned}$$

Assuming the statistical independence of the sources, the model can be simplified in [OG06]:

$$V_{x_i}(t, \omega) := \sum_j |w_{ij}|^2 V_{s_j}(t - \tau_{ij}, \omega) \quad (3.4)$$

3.3. ANECHOIC DEMIXING USING WIGNER-VILLE TRANSFORM 19

As two-dimensional representation of one dimensional signals, this equation is redundant and can be solved by projecting the WVS back to several 1D spaces. The simplest projections are obtained by computing the first and zero-order moments of equation 3.4, which results in the following equations:

$$E\{|\mathcal{F}_{x_i}|^2\} = \int V_{x_i}(t, \omega) dt = \sum_j^n |w_{ij}|^2 \int V_{s_j}(t - \tau_{ij}, \omega) dt = \sum_j^n |w_{ij}|^2 E\{|\mathcal{F}_{s_j}|^2\} \quad (3.5)$$

and:

$$E\{|\mathcal{F}_{x_i}(\omega)|^2\} \cdot t_{x_i}(\omega) = \sum_j^n |w_{ij}|^2 \cdot E\{|\mathcal{F}_{s_i}|^2\} \cdot [t_{s_j}(\omega) + \tau_{ij}] \quad (3.6)$$

Equation 3.5 defines an instantaneous mixing model with non-negativity constraints and can be treated as a standard NMF or positive ICA problem. The unknown sources, the weights and the delays can be estimated by using the iterative solution of a two step algorithm [OG06].

1. Solve:

$$|\mathcal{F}x_i(\omega)|^2 = \sum_j |w_{ij}|^2 |\mathcal{F}s_j|^2(\omega) \quad (3.7)$$

where $\mathcal{F}x_i$ and $\mathcal{F}s_j$ specify the normal Fourier transform of x_i and s_j . This equation can be solved using nonnegative matrix factorization (NMF) or a positive ICA algorithm to obtain the power spectra of the sources.

2. Use the results from the previous step to solve the following equation numerically to obtain the phase spectra and delays of the sources:

$$|\mathcal{F}x_i(\omega)|^2 \frac{\partial}{\partial \omega} \arg\{\mathcal{F}x_i(\omega)\} = \sum_n |w_{ij}|^2 |\mathcal{F}s_j(\omega)|^2 \left[\frac{\partial}{\partial \omega} \arg\{\mathcal{F}s_i(\omega)\} + \tau_{ij} \right] \quad (3.8)$$

In this way, the linear mixture model, as well as the time delays, can be solved using this BSS approach based on the Wigner-Ville transform and has been applied successfully for joint angle trajectories (Chapter 4).

Real-Time Architecture for Character Animation

This chapter describes the realization of a real-time character animation based on a small set of approximated spatial movement primitives, or synergies from full-body motion capture data. In this way, we modeled complex motor behavior with many degrees of freedom by the superposition of simpler components (synergies), inspired by the concepts of motor control (Section 1.3.2). Additionally, we combine the compact parameterization of the motion captured movements with a dynamical system approach to obtain a real-time architecture. This can be realized by constructing a nonlinear mapping between the solutions of the dynamical systems and the extracted source signals by applying supervised learning techniques. To obtain a coordinated behavior, the dynamical systems corresponding to the different source signals are coupled with each other.

4.1 Data Acquisition

4.1.1 Motion Capture System

The Motion Capture System became a common approach to record motions and translating that movement onto a synthetic model. Especially in movies and in video games, motion capturing is a popular process to synthesize realistic human actions before transferring the generated movements onto a virtual character (Figure 4.1).

One can distinguish between two different approaches of motion capturing: The *electromagnetic*, or *magnetic motion capture*, and the *optical motion capture*.

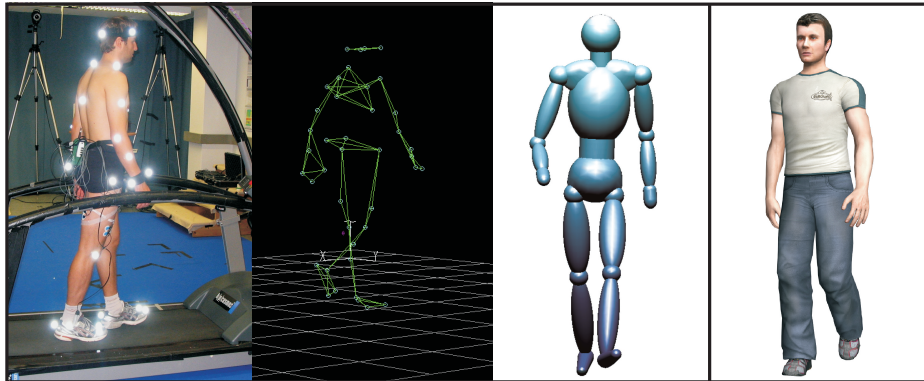


Figure 4.1: From a real performer to a virtual character. Left to right: Capturing human walking; 3D positions of recorded data; joint angle computation and mapping them onto a simple character; and animate a virtual human.

The magnetic tracking method requires an environment devoid of magnetic field distortions and furthermore uses sensors placed at joints. These sensors often need cables to transmit their positions and orientation back to a central processor to record their movements, so that the performer is tethered with some kind of cabling harness, and thus constraining the flexibility of the actor's movements [Par02].

However, the optical motion capturing is a wireless procedure and gives the actor enough flexibility to perform complex movements. This is a very accurate method of capturing motions. It is not a real-time process—at least not for complex movements; immediate feedback is not or hardly possible on the target character. Since the captured data are error prone and noisy, extensive post-processing become necessary (Section 1.3.1.1).

Our lab is equipped with an optical motion capture system, consisting of the VICON 612 data station, which controls nine active infrared cameras with a temporal sampling frequency of 120 Hz and a spatial resolution error of $< 1 : 5\text{mm}$ (Figure 4.1.1). The Vicon motion tracking system includes the software: VICON workstation, VICON Body Builder and the Plug-in Gait marker set.

For our recordings, we used optical markers with a diameter of 2.5cm, which are spheres covered with a retro reflective material. The motion capture cameras are normally fitted with their own light sources that create a directional reflection from the markers. The cameras' threshold can be adjusted, so that only the bright reflective markers will be sampled ignoring skin and fabric.

The optical system must be calibrated by having all the cameras track an object

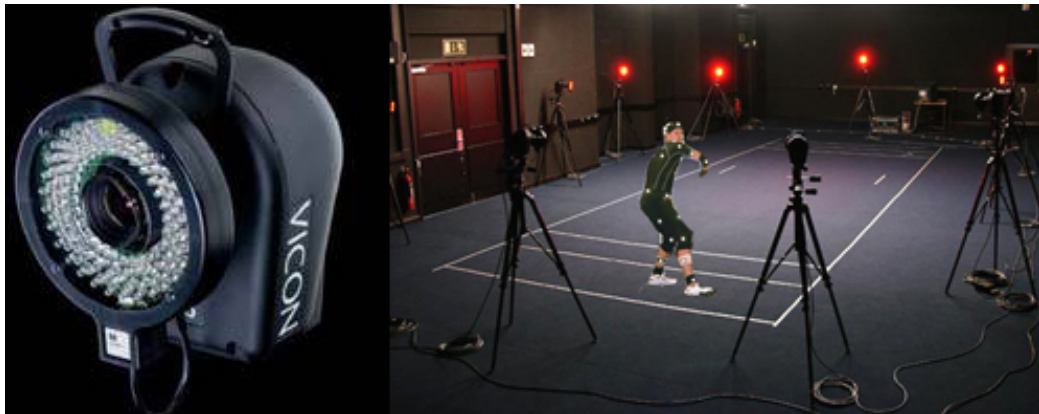


Figure 4.2: Motion capture environment. Vicon cameras record the performer’s motion.

with known dimensions that the software can recognize, such as a wand with reflective markers. By combining the views from all cameras with the known dimensions of the object, the exact position of each camera in space can be calculated. At least two views are needed to track a single point’s three-dimensional position while extra cameras are necessary to maintain a direct line of sight from at least two cameras to every marker [Men99].

For our recordings, 41 optical markers were coated on positions, according to the Plug-in Gait marker placements, listed in the table in Appendix 1, which are taken from the *Plug-in Gait documentation*. Since it is not possible to attach the marker directly to the joint centers of the human skeleton, the coordinates of the joint centers –*virtual markers*– have to be inferred from the position of the skin markers using the Plug-in Gait information and the anatomical measurements of the individual actors, see Figure 4.3. The description how to compute the joint centers from the marker position can be taken from the *Plug-in Gait documentation*. However, these virtual markers then describe the position of the joint centers which are used for the representation of our animated character model, giving the model hierarchy structure of segments which are connected by 17 computed joint centers (Figure 4.4).

4.1.2 Joint Representations of the Avatars

A constant problem with motion capture systems is noise, which can arise, for instance, from the physical system. The markers can move relative to their initial positioning, and the faster the performer moves, the more the marker swing and reposition themselves. Noise also arises from the sampling process. To deal with

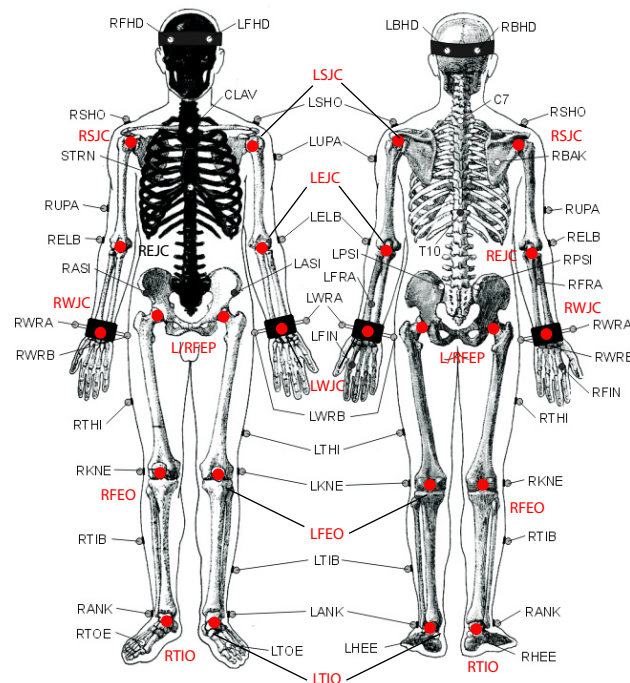


Figure 4.3: According to the Plug-In Gait marker set, gray spheres show the placement of the markers. Red dots indicate the location of virtual markers that can be computed using additional anatomical measurements.

this issue, the user can condition the data before they are used in the reconstruction process. Data points that are radically inconsistent with typical values can be manually thrown out, and the rest can be filtered. The objective is to smooth the data without removing any useful features which is generally handcrafted.

However, once the markers look smooth and reasonable, the next step is to fit the tracking information into a underlying skeletal structure that is to be controlled by the captured motion. In other words, the position of each marker in each frame is used to absolutely position the specific joints of the skeleton.

4.1.2.1 Joint angle computation

The cleaned and post-processed motion capture data describe the 3D-position of the virtual markers during human motions. Because the comparison between the movements with absolute marker positions are difficult to analyze and the major part of the variance will be dominated by the actor's height differences, the movement is described by the angular change between segments.

To compute the joint angles, we use a forward kinematic model. Its essential con-

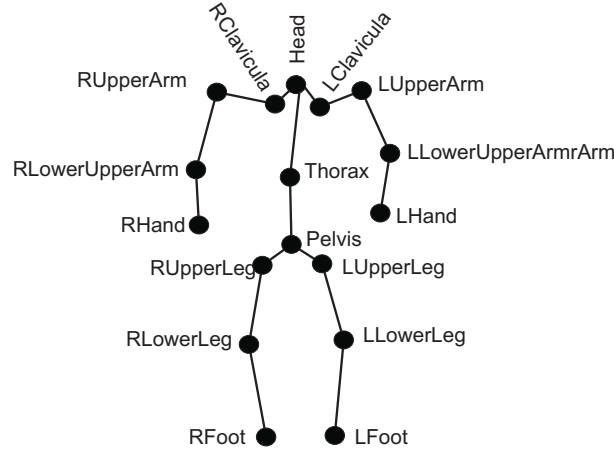


Figure 4.4: Representation of the hierarchical structure of our model.

cept is that the position of the end effector of the model is calculated by taking into account the position and orientation of the previous segments in the kinematic chain (forward kinematic). The relative orientation between adjacent segments is given by a rotation matrix mapping one reference frame to the other. A complete list of segments, the associated markers, the joints, and the corresponding trihedron can be found in Table 2.

The standard angle computation of joints are based on Euler angles [Par02] but also implies certain problems. For instance, one is known as the *gimbal lock* and describes the loss of one degree of freedom that occurs when two axes of the three gimbals are driven in the same place and one degree of freedom is lost. The following example illustrates this:

Rotations around the cartesian axes are given by

$$\mathbf{Z} = \begin{pmatrix} \cos \phi & \sin \phi & 0 \\ -\sin \phi & \cos \phi & 0 \\ 0 & 0 & 1 \end{pmatrix}, \mathbf{Y} = \begin{pmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{pmatrix}, \mathbf{X} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos \psi & \sin \psi \\ 0 & -\sin \psi & \cos \psi \end{pmatrix}$$

Any rotation matrix R in 3D space can be represented by three rotations in several ways. One of these representations is:

$$R = \mathbf{Z}\mathbf{X}\mathbf{Z} = \begin{pmatrix} \cos \phi & \sin \phi & 0 \\ -\sin \phi & \cos \phi & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos \psi & \sin \psi \\ 0 & -\sin \psi & \cos \psi \end{pmatrix} \begin{pmatrix} \cos \psi & \sin \psi & 0 \\ -\sin \psi & \cos \psi & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

with ϕ and ψ constrained in the interval $[-\pi, \pi]$, and θ constrained in the interval $[0, \pi]$. If $\theta = 0$ the above expression becomes equal to:

$$R = \begin{pmatrix} \cos \phi & \sin \phi & 0 \\ -\sin \phi & \cos \phi & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos \psi & \sin \psi & 0 \\ -\sin \psi & \cos \psi & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

The second matrix is the identity matrix and has no effect on the product. Carrying out matrix multiplication of the first and the third matrices, one computes:

$$R = \begin{pmatrix} \cos(\phi + \psi) & -\sin(\phi + \psi) & 0 \\ \sin(\phi + \psi) & \cos(\phi + \psi) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Changing ϕ 's and ψ 's values in the above matrix has the same effect. The rotation's angle $\phi + \psi$ changes, but the last column and the last line in the matrix will not change. Hence, the rotation's axis remains in the Z direction and one degree of freedom has been lost.

One can choose another convention for representing a rotation with a matrix using Euler angles than the $Z - X - Z$ convention above, and also choose other variation intervals for the angles, but at the end there is always at least one value for which a degree of freedom is lost. Therefore, our joint angle computation is based on the *axis angles* representation.

4.1.2.2 Axis Angle

The axis angle representation evolves from Euler's rotation theorem and implies that any rotation of a rigid body in a three-dimensional space is equivalent to a pure rotation around a single fixed axis. Hence, the axis angle represents a rotation by two values: A unit vector indicating the direction of a directed axis, and an angle describing the magnitude of the rotation about the axis.

Mathematically the connection between the axis of rotation ϵ , the angle of revolution θ and the rotation matrix R is given by:

$$\hat{\epsilon} := \begin{pmatrix} 0 & -\epsilon_3 & \epsilon_2 \\ \epsilon_3 & 0 & -\epsilon_1 \\ -\epsilon_2 & \epsilon_1 & 0 \end{pmatrix}$$

$$R = I + \sin \theta \hat{\epsilon} + (1 - \cos \theta) \hat{\epsilon}^2$$

where R is a 3×3 rotation matrix and the hat operator gives the antisymmetric matrix equivalent of the cross product. This is known as the Rodrigues' rotation formula.

To obtain the axis angle representation of a rotation matrix, the angle of rotation has to be computed and then has to be used to find the normalized axis:

$$\theta = \arccos\left(\frac{\text{trace}(R) - 1}{2}\right)$$

$$\epsilon = \frac{1}{\sqrt{(R_{32} - R_{23})^2 + (R_{13} - R_{31})^2 + (R_{21} - R_{12})^2}} \begin{pmatrix} R_{32} - R_{23} \\ R_{13} - R_{31} \\ R_{21} - R_{12} \end{pmatrix}$$

The three parameter of the axis angle parametrization are then defined as the entries of the vector θw with length θ and direction ϵ . Only the norm of this representation is affected by the 2π angle ambiguity, and in practice this leads only to flips in sign of the vector $\theta\epsilon$ which are easy to remove [Oml10].

4.2 Data Reduction Based on BSS

Synthesizing natural-looking animations of human, learning-based approaches have become increasingly popular. The efficient parameterization of complex movements is a core problem in modern computer animation caused by the degrees of freedom problem (Section 1.3.2). The size of the required databases depends critically on the efficiency of the chosen trajectory representation. One of its characteristics is that such representations should allow the synthesis of large trajectory classes with a very limited amount of motion capture data. This can be achieved by obtaining a lower dimensional trajectory representation using unsupervised learning algorithms, which became a widespread method as a result.

For an accurate re-synthesis of the trajectories standard, methods for dimension reduction, such as PCA or ICA, nevertheless require about 8 – 12 components for an accurate approximation of human full-body movements [SHP04]. To develop efficient representations of motion data, we take an approach of blind source separation (Section 3), which promises a significantly more compact trajectory representation.

4.2.1 Database of Recorded Data

The captured and post-processed movement trajectories were retargeted to a skeleton model with 17 joints using an axis-angle representation (Section 4.1.2.2) for the parameterization of the 3D rotations between adjacent segments as discussed

above. The recorded database comprises different types of periodic gaits. Partially the gaits are combined with non-periodic arm movements, such as swinging the right or left arm up, or holding the arm in a fixed posture during the whole gait cycle. For these recorded movements, the arm moved independently from the legs, forming a separate 'synergy' that was non-periodic. Moreover, we captured crouching and straight walking with neutral and emotional styles as happy, angry, fearful and sad. Additionally, walking along a circular path (forward and backward) with rotations of 90 degrees, and turning on spots (120 deg) left or right per double step were recorded for navigation purposes.

4.2.2 Learning Movement Primitives

It was possible to learn compact movement primitives from our dataset described above in Section 4.2.1, containing periodic and non-periodic motions by applying the blind source separation approach, based on the anechoic mixture model, which was mentioned before in Section (Section 3.3). As a result, a highly compact data representation could have been obtained.

Before the algorithm was applied to the data, the joint angles x were preprocessed by *centering* them through subtracting its means m . In this way, it is ensured that the data have a zero-mean, which results in a zero average. This is done solely to simplify the BSS algorithm, which does not mean that it could not be approximated. After subtracting the means m_i the joint angle trajectories x_i were approximated by a weighted mixture of source signals, that are given by the equation:

$$x_i(t) = m_i + \sum_j w_{ij} s_j(t - \tau_{ij}) \quad (4.1)$$

The functions s_j denote the hidden source signals while the parameters w_{ij} are the mixing weights. As opposed to common blind source separation techniques like PCA or ICA, this mixing model allows for time shifts τ_{ij} of the sources in the linear superposition. Source signals, time shifts or delays, and mixing weights are determined by a blind source separation algorithm that is based on a time-frequency integral transform [OG06].

Moreover, the source functions are joint specific, but identical over the whole data set in contrast to the mixing weights and delays. A detailed analysis shows that mixing weights and delays vary with different motion styles. In other words, for the generation of intermediate or different motion styles the weights and delays for each motion style are required. By interpolating or blending between them, the desired behavior can be modified.

In this way, we estimated a common set of three or four source functions which

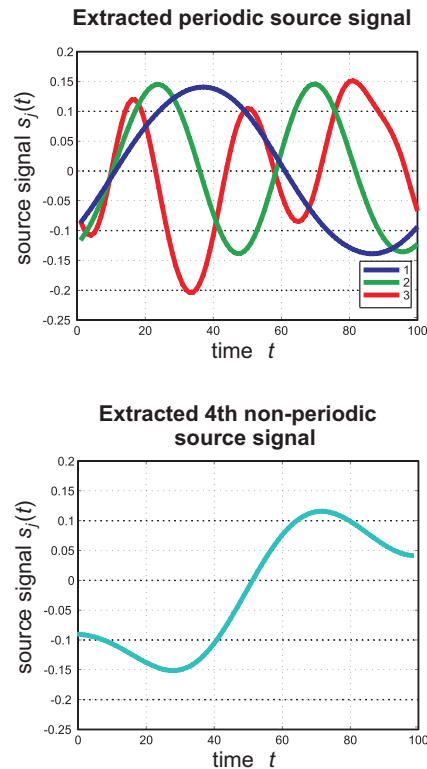


Figure 4.5: Four source signals were extracted from the joint angles trajectories of the motion capture dataset. Sources 1 – 3 represent the periodic and the 4th source the non-periodic movements.

explain the whole motion data. An example is given in Figure 4.2.2: Four approximated source signals from the joint angle trajectories of the different periodic and non-periodic actions are illustrated, where three of them describe the periodic and one describes the one shot movements.

4.2.2.1 Approximation Quality

Detailed comparisons of periodic and non-periodic trajectory data show that the applied anechoic mixture model provides a more compact approximation of human movement trajectories and requires less source terms than models based on instantaneous mixtures. This is illustrated in Figure 4.2.2.1 for the described data set. The figure shows the approximation quality as a function of the number of sources s_j . Instead of the explained variance $1 - (\|X-F\|_F/\|X\|_F)^2$, we used a

quality measure Q that is more sensitive for differences in the regime of small approximation errors. It was given by the expression:

$$Q := 1 - \frac{\|X - O\|_F}{\|X\|_F}$$

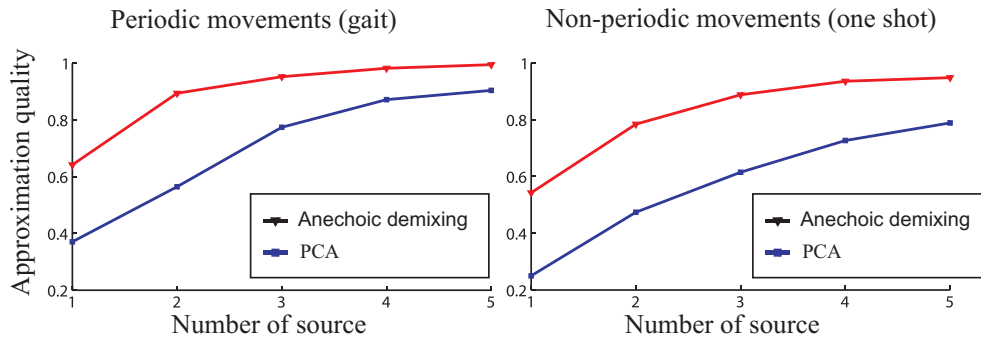


Figure 4.6: Comparison of different blind source separation algorithms of periodic and non-periodic movements. The approximation quality Q is shown as a function of the number of sources for traditional blind source separation algorithm (PCA) and the applied new algorithm.

where X signifies the original data matrix and O its approximation by the source model. The norm is defined as the Frobenius norm.

The approximation quality of the recorded trajectory set including periodic and non-periodic movements with only four source signals is $Q = 0.92$. This corresponds to an explained variance of approximation. $E = 0.99$ and is sufficient for a very accurate approximation of the trajectories. The traditional blind source separation algorithm, like PCA and ICA, requires more than seven sources to achieve the same level of accuracy. Regarding only periodic gait movements, only three source functions were sufficient to obtain a highly accurate estimation of the joint trajectories where PCA and ICA need twice the amount of the sources to obtain the same approximation quality (Figure 4.2.2.1).

4.3 Dynamical System for Trajectory Generation in Real-Time

The described model 4.1 for the compact movement representation is not yet suitable for real-time application as it is needed for interactive animations, e.g. in

video games. For an immediate process, the joint trajectories have to be synthesized by superposition and delaying of source signals, whose whole time-course must be known. This leads us to the question: How do we transfer or express the learned components into a real-time capable architecture, which provides the joint angles trajectories iteratively? A real-time capable system can be devised by specifying dynamical systems that produce the same shape of the extracted movement primitives (Figure 4.7) as a solution.

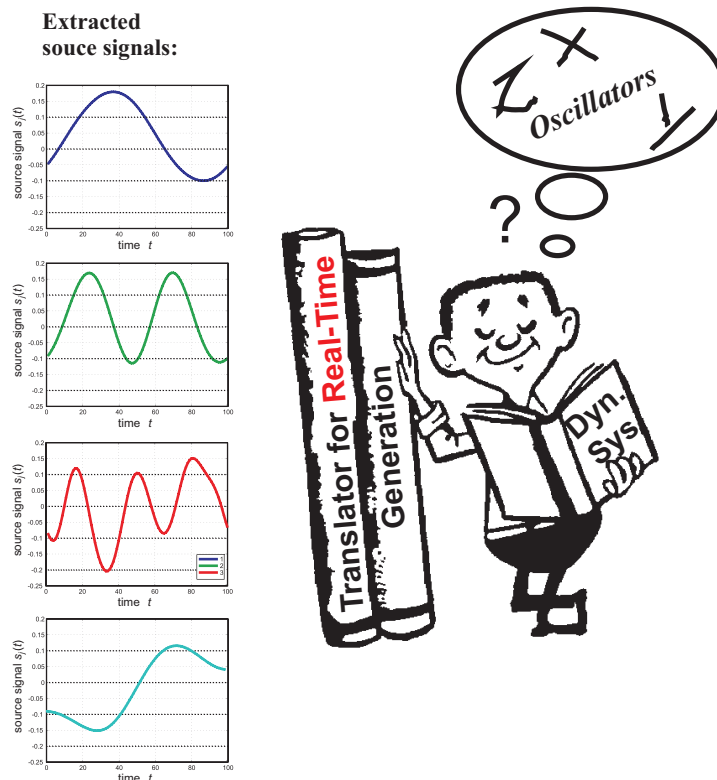


Figure 4.7: How can the extracted source signals be encoded for a real-time application?

We design such dynamical systems again by exploiting the fact that the movements can be approximated by a superposition of a few basic components. For this purpose, we learn the shape of the components by establishing a mapping between the solutions of simple dynamical systems and the source signals of the trajectory representation. In this way, the complete trajectory can be generated by the superposition of the solutions of a set of such simple dynamical systems.

4.3.1 Attractor Dynamics

The four extracted source-signals are sufficient for a highly accurate approximation of the original trajectories, containing periodic and non-periodic movements, as we have seen before in Section 4.2.2.1. The fourth source contributes exclusively to the reproduction of the non-periodic movements in the data set, whereas the other three sources are periodic and model the periodic parts of the movements.

The idea for the online generation of the trajectories is to construct dynamical systems that generate the source signals $s_j(t)$ by iteration of differential equations over time. The relative timing between the different source signals can be stabilized by introducing dynamic couplings between these differential equations, which will be discussed later in Section 4.3.4.

However, efficient realization of the time delays presents an additional challenge for the online implementation because introducing them would lead to a slow system dynamics with rather complex stability properties. This particularly is a problem for more complex systems including multiple interacting characters.

In the following section, we first introduce the differential equations that generate the source signals in an online fashion (Section 4.3.2). Therefore, we have to choose structurally stable nonlinear dynamical systems for the generation of periodic and non-periodic patterns, and then map their solutions onto the required form of the source signals –applying kernel methods (Section 4.3.3).

4.3.2 Dynamic Primitives

For the online generation of the motion trajectories, a nonlinear mapping between the solutions of the dynamical systems and the estimated source signals is constructed. This gives us high flexibility for choosing dynamical systems, which can be optimized in order to simplify the design of a stable overall system dynamics. As basic building blocks for the dynamics, nonlinear oscillators are used to synthesize periodic behaviors, while a fixed point attractor dynamics is used for the synthesis of non-periodic movements.

We decided to make use of nonlinear dynamical systems, whose structural properties do not change in the presence of weak couplings. In this way, we can design the qualitative properties of different dynamic primitives independently from their interaction with other system components.

The idea to map desired behaviors onto solutions of nonlinear dynamical systems is quite common in robotics and behavioral research and has been successfully demonstrated in a variety of applications [SDE95, INS02, SIB03, BRI06], as well

as in computer animation [FMJ02].

It seems natural to choose limit cycle oscillators as dynamics for the generation of such periodic behaviors, at least for periodic movements (e.g. [INS02]). Because of its well-studied dynamics, we use the Van der Pol oscillator as a base element of our architecture for the generation of the periodic signals. For an adequate choice of the parameters, the Van der Pol oscillator (described in Section 2.2.3) produces an asymptotically stable limit cycle and its dynamics is given by the differential equation:

$$\ddot{y}(t) + \zeta (y(t)^2 - a) \dot{y}(t) + \omega_0^2 y(t) = 0 \quad (4.2)$$

The parameter ω_0 determines the eigenfrequency of the oscillator, and the positive parameter a the amplitude of the stable limit cycle. Given an appropriate choice of the oscillator parameters, the force determined by the parameter $\zeta > 0$ pushes, after a perturbation, the state back towards the limit cycle in absence of external input signals. Assuming appropriate scaling of the two axes the form of the stable limit cycle can be made almost ideally circular in the y - \dot{y} plane (phase plane). This property is needed for the online implementation of the phase delays, which will be discussed later in Section 4.3.3.

The fourth source (see Figure 4.2.2) belongs to the one shot motion and is crucial for the approximation of the non-periodic arm movements, for which the arm moves from a start posture to an end posture. We model this behavior by a fixed point dynamics. Considering that natural arm movements are characterized by a bell-shaped velocity profile [SL81, Mor81], we chose a nonlinear dynamics that generates solutions with this property, and which can generate identical movements in opposite directions by the change of a single parameter. This dynamics is given by the differential equation:

$$\dot{y}(t) = uy(t)(1 - y(t)) \quad (4.3)$$

We restrict the values for y to the interval $I = [0, 1]$. For $u < 0$ this dynamics has a stable fixed point in 0, and for $u > 0$ a stable fixed point in 1 that is approached asymptotically from inside the interval I . The value of $|u|$ determines how fast this fixed point is approached. The solution of this differential equation can be computed analytically and is given by

$$y(t) = (1 + \tanh(\frac{u}{2}(t - t_0)))/2, \quad (4.4)$$

showing that its derivative

$$\dot{y}(t) = \frac{u}{4}(1 - \tanh^2(\frac{u}{2}(t - t_0))) \quad (4.5)$$

is bell-shaped. By clipping we ensure that in presence of noise y does not leave the permissible interval.

4.3.3 Mapping between Attractor Solutions and Source Signals

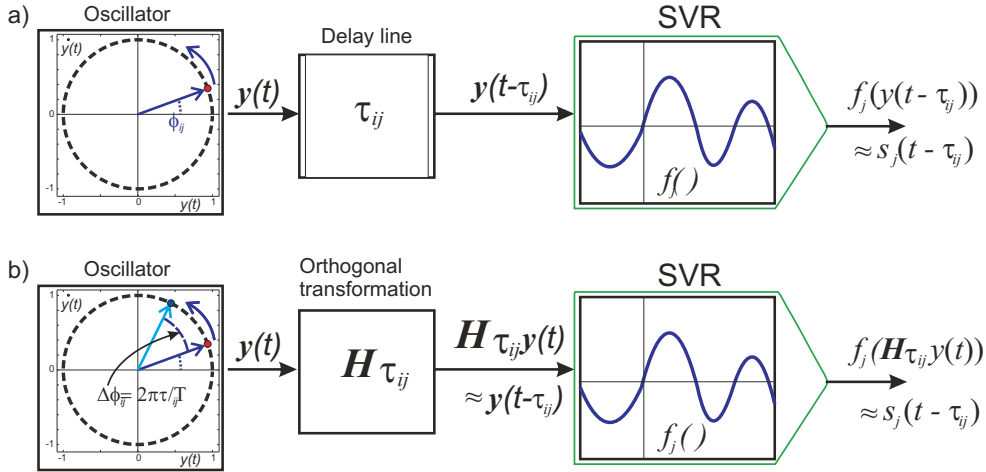


Figure 4.8: Online implementation of delays. a) Direct implementation introduces explicit delay lines, resulting in a complex system dynamics that is difficult to control. b) Approximation by a rotation in the phase space, defined by the instantaneous orthogonal transformation $\mathbf{H}_{\tau_{ij}}$ in the phase plane of the oscillator, avoids a dynamics with delays.

To realize an iterative trajectory generation and to achieve a real-time capable system architecture, a dynamical system is needed to reproduce the same shape or form of the extracted movement primitives. In this way, we can replace the learned sources by these dynamical systems that reproduce the sources signals in real-time and can be defined as dynamic primitives. Therefore, we construct a nonlinear mapping between the attractor solutions of the differential equations and the estimated components. This mapping is defined by a concatenation of a rotation in phase space, modeling the influence of the time delays τ_{ij} , and a nonlinear function, which is learned from training data points by *Support Vector Regression* (SVR). The underlying idea is illustrated in Figure 4.9.

By treating the oscillatory primitives first, the purpose of the mapping is to associate the points $\mathbf{y} = [y, \dot{y}]'$ along the attractor in the phase plane of the Van der Pol oscillators with the corresponding values of the source function s_j . We try to avoid the introduction of explicit time delays in the implementation since this would lead to a complex system dynamics. Instead, as illustrated in Figure 4.8,

we approximate the terms $s_j(t - \tau_{ij})$ in (4.1) in the form:

$$s_j(t - \tau_{ij}(t)) \approx f_j(\mathbf{H}_{\tau_{ij}}\mathbf{y}(t)) \quad (4.6)$$

where $\mathbf{H}_{\tau_{ij}}$ is an affine transformation of the form:

$$\mathbf{H}_{\tau_{ij}} = \begin{bmatrix} \cos(\phi_{ij}) & -\sin(\phi_{ij}) \\ \sin(\phi_{ij}) & \cos(\phi_{ij}) \end{bmatrix} \Sigma \quad (4.7)$$

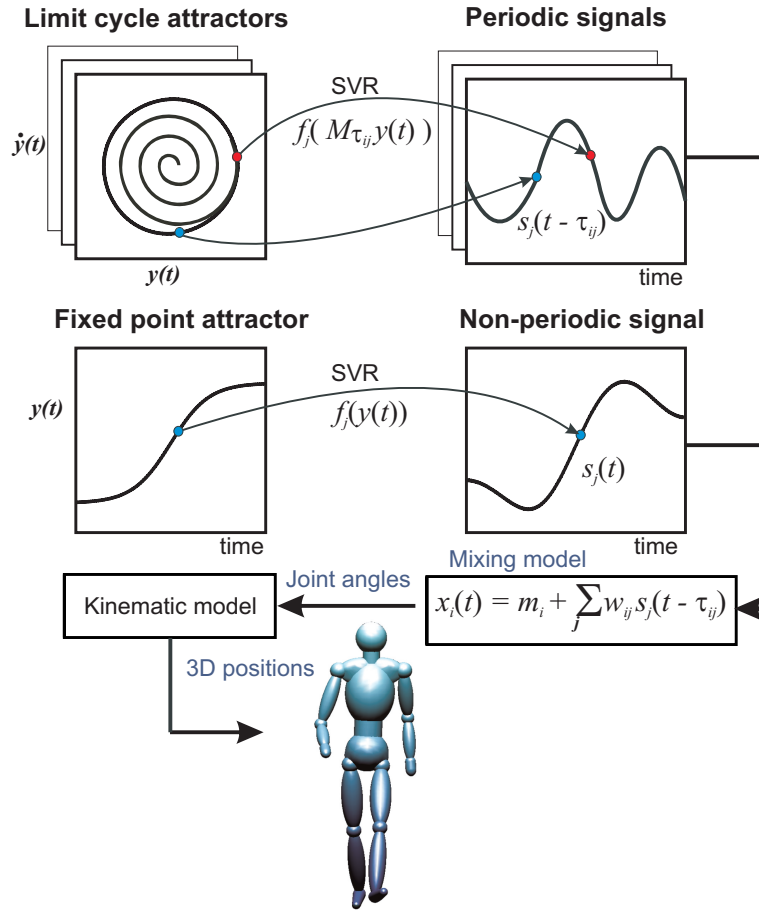


Figure 4.9: Illustration of the dynamic architecture for real-time animation. Periodic and non-periodic movements are generated by dynamic primitives. The solutions of these dynamical systems are mapped onto the source signals by a nonlinear mapping that models the time delays and a nonlinear transformation that is learned by SVR. Joint angle trajectories can be synthesized by combining the signals linearly according to the learned mixture model 4.1. A kinematic model converts the joint angles into 3-D positions for animation.

This transformation is a concatenation of scaling and rotation in the two dimensional phase space. The matrix Σ is diagonal and scales the axes of the phase plane in a way that makes the attractor solution approximately circular. The rotation angles are given by $\phi_{ij} = -2\pi \frac{\tau_{ij}}{T}$, where T is the duration of one period of the stable oscillatory solution.

However, the nonlinear function $f_j(\mathbf{y})$ maps the phase plane onto a scalar. It is learned from training data pairs that were obtained by temporally equidistant sampling of the signals $\mathbf{y}(t)$ and $s_j(t)$, where we used the solutions of the uncoupled dynamic primitives. The functions were learned by support vector regression [Vap98, CL01] using a Gaussian kernel.

In order to learn the form of the non-periodic source signal, we mapped the solution of the point attractor (equation 4.3) onto the values of the non-periodic source signal. Therefore, we used the same supervised learning method as for observing the periodic joint trajectories. In principle, here the effect of the time delay can be modeled by an application of a conformal mapping to the solution of this equation. The overall system dynamics was defined by three Van der Pol oscillators and the point attractor dynamics. The state variables of the dynamic primitives were mapped onto the source signals by the described nonlinear observers.

After all, we were able to transfer the most compact motion representation, consisting of complex kinematics, into simple nonlinear dynamical primitives. Furthermore, the dynamical primitives can be generated in real-time. We then observe the complete joint angle representation by the linear combination, according to the mixture model 4.1 for each iteration step. This step requires the consideration of the average joint angles m_i , which were subtracted before –see Section 4.2.2– and the style dependent mixing weights w_{ij} . Afterwards, the synthesized movement trajectories can be transferred to a kinematical model.

With the presented system architecture, the online character animation can be controlled by adjustments of few parameters of the nonlinear dynamical systems. The reason is caused by the fact that the change of the dynamical systems have an immediate effect to the corresponding output signals. As a consequence, we observe the modified joint angle trajectories in an online fashion.

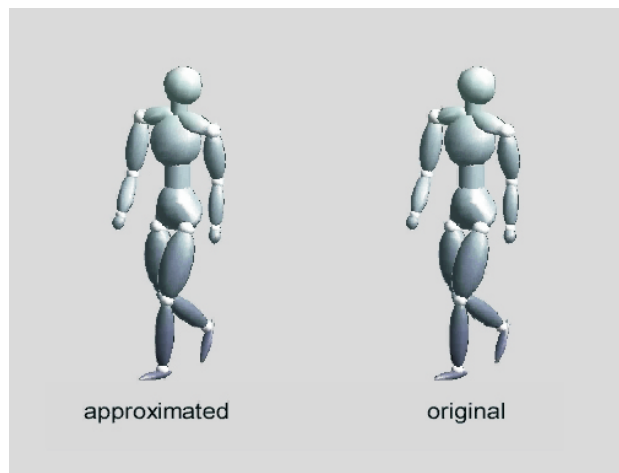


Figure 4.10: Demo simulates the comparison between approximated and original motion. Furthermore, the comparison between different emotions can be seen in the following movies: **MovieAngry**, **MovieSad**, **MovieHappy**.

Beyond the quantitative analysis in Figure 4.2.2.1, the influence of approximation quality on animations is illustrated in the attached demos in Figure 4.10 and shows the comparison between the original and the approximated gait animation, which were generated by the dynamical system. An overview of the whole algorithm is given in the Figure 4.9.

Reconstruction Error

The blind source separation method discussed in Section 3 provides accurate approximations of the joint trajectories by using less components than other standard methods for dimension reduction as we have seen in Section above.

As further evaluation of the method, we compared the reconstruction errors of the model for different conditions, including online and offline models. For this purpose, we extracted three shift invariant source signals from the seven motion-captured emotional walking gaits (including left and right turning walks).

We compared the reconstructed angle trajectories of neutral straight walking of synthesized gaits with the original motion-captured trajectories of the same style, corresponding to avatar 1 (A_1). The second case was the offline generation of the trajectory by the blind source separation model (Chapter 3), which was used to animate avatar 2 (A_2). The movements of avatar 3 (A_3) were generated by the simplest possible online model, mapping the phase spaces of three coupled oscillators by Support Vector Regression onto the source signals. The propagation speed of this avatar deviates slightly (a few percent) from the avatar 1 due to approximation errors.

The normalized errors (unexplained variance) between the trajectories in joint angle space of A_2 and A_3 compared with A_1 (original trajectory) were: 6.89% ($A_2 - A_1$), and 8.62% ($A_3 - A_1$).

This implies that the major error is introduced by the source approximation, while the support vector regression has a much smaller influence.

4.3.4 Stabilization by Dynamic Coupling

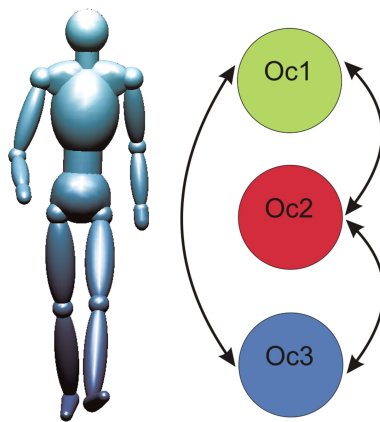


Figure 4.11: Every avatar is driven by three limit cycle oscillators. The coupling of the oscillators ensure coordinated behavior and prevent the influence of external perturbations by dynamical noise.

As indicated by the system architecture, every avatar is driven by three limit cycle oscillators (Figure 4.11), where each of them represents one movement primitive as its output. We introduce dynamic couplings between the primitives to obtain a stable and a more robust behavior against potential interference from the environment. Hence, the temporal coordination between the different source components can be ensured while the modulation of the locomotion in terms of simple control signals can be designed. One counterexample is presented in Figure 4.12, which shows the result of uncoupled oscillators or dynamic primitives, resulting in an uncoordinated behavior. The figure and the corresponding demo show two characters next to each other, where each of them is driven by three, either coupled (right side) or not-coupled (left side) oscillators representing the three corresponding movement primitives (Section 4.3.3). To prevent eventual disturbances from e.g. the environment, perturbation was simulated by adding dynamic noise v into one of the oscillatory primitives (VdP oscillators). Within a short period of time, we observe that the system recovers from the perturbation in

case of coupled oscillators, whereas when not coupled the behavior of the system remains uncoordinated.

$$\ddot{y}(t) + \zeta (y(t)^2 - a) \dot{y}(t) + \omega_0^2 y(t) + v = 0 \quad (4.8)$$

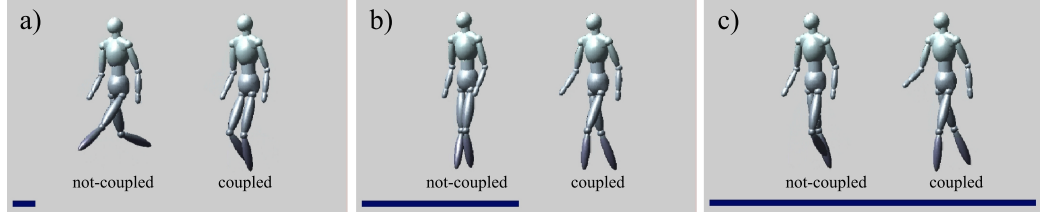


Figure 4.12: Comparison between two characters which are consisting of *uncoupled* and *coupled* dynamic primitives. a) Dynamic noise is included, in order to simulate e.g. disturbances or environmental noise. b), c) While the movement of the left character becomes uncoordinated, the coupled character on the other hand gets stabilized. The corresponding movie can be found under **DEMO**.

The coordination or synchronization can be easily accomplished by introducing couplings between the dynamic primitives: Applying concepts from *contraction theory* (explained in Section 2.3) can guarantee a single stable solution for even complex networks of such limit cycle oscillators if the oscillators are coupled in terms of *velocity couplings*. This type of coupling is defined by the equations (k specifying the coupling force):

$$\begin{aligned} \ddot{y}_1 + \zeta (y_1^2 - a) \dot{y}_1 + \omega_0^2 y_1 &= k (\dot{y}_2 - \dot{y}_1) + k (\dot{y}_3 - \dot{y}_1) \\ \ddot{y}_2 + \zeta (y_2^2 - a) \dot{y}_2 + \omega_0^2 y_2 &= k (\dot{y}_1 - \dot{y}_2) + k (\dot{y}_3 - \dot{y}_2) \\ \ddot{y}_3 + \zeta (y_3^2 - a) \dot{y}_3 + \omega_0^2 y_3 &= k (\dot{y}_1 - \dot{y}_3) + k (\dot{y}_2 - \dot{y}_3) \end{aligned} \quad (4.9)$$

For values of α below a specific bound, which depend on the coupling graph, the overall system dynamics has only one single stable solution. It is characterized by synchronization of all oscillators.

To synchronize the non-periodic primitives with external events, the sign of the parameter u in equation 4.3, was switched depending on an external signal, which triggers one shot movements such as the raising of the arm. In this way, the previously stable fixed point of this dynamics becomes unstable while its unstable fixed point becomes stable. In addition to this, we added a short pulse input to this equation that displaces the state away from the unstable fixed point. This ensures a transition to the novel stable point with a well-defined timing.

CHAPTER 5

Dynamic Character Animation



Figure 5.1: Virtual humans present a standard element in most modern entertainment software.

Almost in every video game, animated human motions are involved. Today, gamers find themselves in more realistic computer graphic worlds competing in increasingly complex missions with more believable characters. Not only in video games character animations are important, but virtual humans present a standard

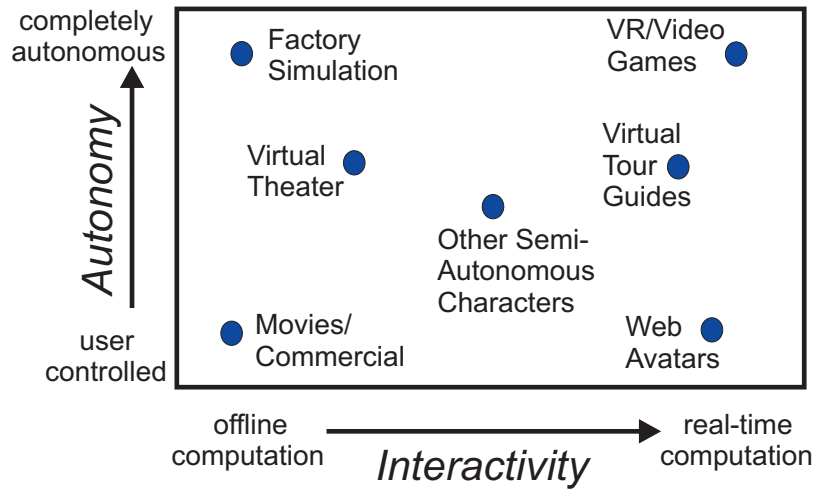


Figure 5.2: This figure present examples of the relationships between autonomous and interactive character animations

element in every modern entertainment software (see Figure 5). Major airlines, for instance, are using avatars to replace stewardesses to present in-flight security videos; crowds are simulated for animating dangerous scenes on most Hollywood films, etc. However, a scene is only as intriguing as the characters that inhabit its world. This chapter introduces the methods we apply for obtaining an interactive behavior, which can be combined for self-organized crowd simulations.

5.1 Autonomous Agents for Interactive Character Animation

The degree of autonomy depends on the intervention of the user. The less the characters need instructions by the user to react in a specific way or interact with the dynamic environment, the more independent and autonomous the avatars are. In movies, for instance, the user fully controls and synthesizes the joints of a character while working offline. Figure 5.2 shows diverse applications of character animations and explains their levels of autonomy and interactivity. In this context, the interactivity represents the reaction time the character efforts to interact with his surroundings. As it is evidenced in movies, no strict execution time is required for modeling human animations. In other applications, particularly in computer games, interactivity in real-time has a high relevance.



Figure 5.3: Depending on specific weight components, different motion styles can be simulated. Example: Boy walking sadly on the left side and happy on the right side.

Despite the challenging task of obtaining a fully-autonomous human character motion for applications that provide real-time interaction, we are able to animate self-organized autonomous character animation based on the learned dynamic primitives. In this section, we will introduce the first steps towards autonomy, which of:

- Generating different motion styles,
- character propagation, and
- navigation and path planning

5.1.1 Dynamic Style Morphing

The proposed framework for the real-time synthesis of human movements can be integrated with other functions that are crucial for applications with autonomous characters. We discuss in this section the integration of style morphing within this framework. Our dataset (Section 4.2.1) consists of periodic gait movements with different emotional styles, direction changes as well as simple on-shot movements. A distinction between the different motion styles can be made due to the specific weight components for individual styles. Hence, to obtain e.g. a sad walking, the movement primitives are combined with the weight components corresponding to its sad style. Two examples are given for sad and happy walk in the

movies in Figure 5.3.

As mentioned in Section 4.3.3, we observe a real-time generation by a linear combination of the dynamic primitives s_{ij} , average angles m_i and the mixing weights w_{ij} , whereby the sources are constant for all styles. The real-time generation of trajectories permits style morphing by using linear interpolation of the mentioned parameters. The interpolation of the mean angles and of the weight matrices is straight forward.

However, the interpolation of the delays requires an additional approximation to avoid artifacts that are caused by ambiguities in the estimation of the delays.

For periodic source signals, ambiguities in the estimation of weights and delays can arise and need to be removed prior to the interpolation. Periodic source signals fulfill

$$s_j(t + rT) = s_j(t) \quad (5.1)$$

with integer r . Furthermore, source signals can be approximately periodic, fulfilling

$$s_j(t + qT_n) \simeq s_j(t) \quad (5.2)$$

with integer q . Such ambiguities are a particular problem for source signals that model the higher frequency components, in which $T_n = T/n$ is an integer fraction of the gait cycle time T . This approximated periodicity can cause ambiguities in the estimated delays, which might differ by multiples of T_n . If such delays are linearly interpolated, they then introduce phase differences between the sources that do not interpolate correctly between similar motion styles.

To remove such ambiguities, we introduced an additional approximation step and we replaced the delays by the modified source delays

$$\tilde{\tau}_{ij} = \tau_{ij} - qT/n \quad (5.3)$$

in which q was chosen to minimize the values of the delays. This approximation was based on an algorithm that identifies the presence of ambiguities by determining the local extrema of the cross correlation function between the original and the time shifted versions of the source signals. This made it possible to restrict and interpolate the delays within the intervals $[-T/2n, T/2n]$, removing the ambiguity. After estimating these modified delays the mixing weights were re-estimated to optimize the accuracy of the obtained model.

By combining the corrected time delays $\tilde{\tau}_{ij}$ and mixing weights, which belong to only a small repertoire of different motions, we are able to obtain a huge variation

of movement styles (Figure 5.5). The following example shows a simple interpolation between two movement styles (a) and (b), e.g. neutral and emotional walking, see also Demo 5.4, and is characterized by the equations:

$$\begin{aligned} m_i(t) &= \eta(t) m_i^a + (1 - \eta(t)) m_i^b \\ w_{ij}(t) &= \eta(t) w_{ij}^a + (1 - \eta(t)) w_{ij}^b \\ \tilde{\tau}_{ij}(t) &= \eta(t) \tilde{\tau}_{ij}^a + (1 - \eta(t)) \tilde{\tau}_{ij}^b \end{aligned} \quad (5.4)$$

The time-dependent morphing parameter $\eta(t)$ specifies the movement style and, as a result, the desired motion can be achieved dynamically. To model transitions

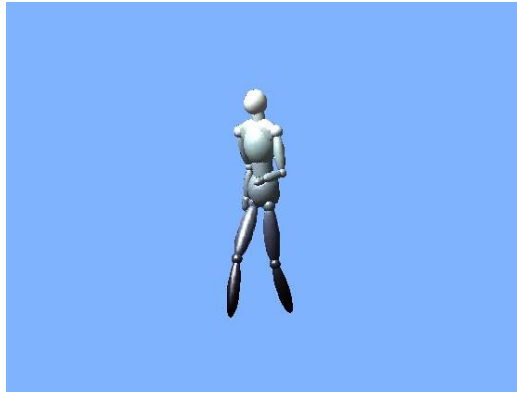


Figure 5.4: Morphing between neutral and happy walk.

between periodic and non-periodic movements, such as walking while lifting up the arms vs. walking with an arm up or an arm down, we used a non-periodic ramp-like source signal to model changes in the mean angles before and after the transition. On the one hand, the superposition weights from periodic and the non-periodic sources are thereby learned during the middle of the transition of the arm movement from training examples. Conversely, the weights for periodic movements are learned from the whole gait cycle, but because the one shot function does not allow shifts and leads to jumps in the trajectories, the weights from the middle of the transition period are learned.

However, by using the equations 5.4, the linear interpolation generates a blending between the periodic gait steps and the steps containing the non-periodic transitions. While applying this method, we were able to generate natural-looking transitions even for movements dependent on periodic and non-periodic primitives.

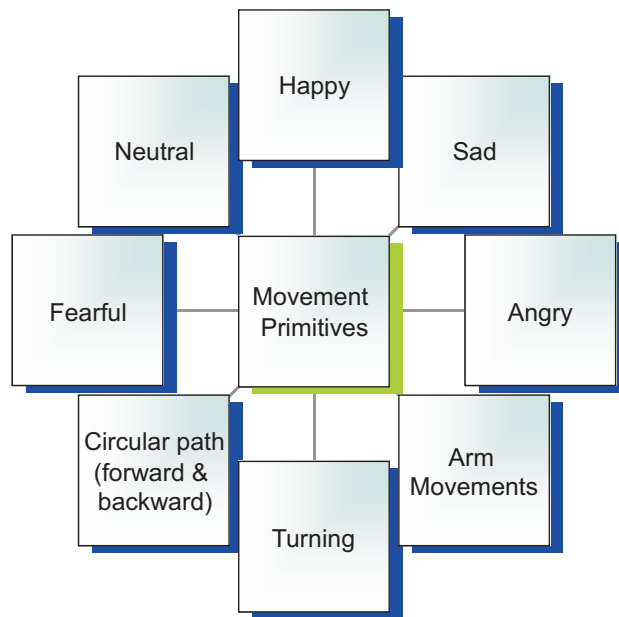


Figure 5.5: Movement primitives can be combined with specific weight components for individual styles. By morphing between them, a huge repertoire of different movement styles is given.

5.1.1.1 Integration of Periodic and Non-Periodic Synergies

The algorithm for online style morphing, discussed above in Section 5.1.1, was tested by applying it to a complex sequence of locomotion patterns. The movements were generated by interpolation between five prototypical gaits in our data set: Straight walking with neutral and happy emotion, rotation steps of backwards and forward walks, and walking with stooped posture and turning on the spot. Although these types of locomotion were quite different, we were capable of approximating them with only three different source terms. By applying the proposed technique for the interpolation between weights, posture vectors, and time delays, we were able to create almost-realistic transitions between these different patterns, resulting in a complex sequence of steps that could be part, for instance, of a dancing scenario.

In this context, we tested that our method correctly identifies the spatial localization of periodic and non-periodic motion components in the training data. The mixing weights w_{ij} for the fourth non-periodic source (cf. Figure 4.2.2) are significantly different from zero only for the angles of the shoulder and elbow joint, reflecting the fact that in our data set the non-periodic movements were mainly executed with the arms. The separation of different spatially localized movement

components makes it possible to modify the movement styles of different synergies separately. This is particularly true for periodic and non-periodic primitives, resulting in the generation of novel movement patterns when combined with such primitives in ways that were not present in the training data. A simple example is

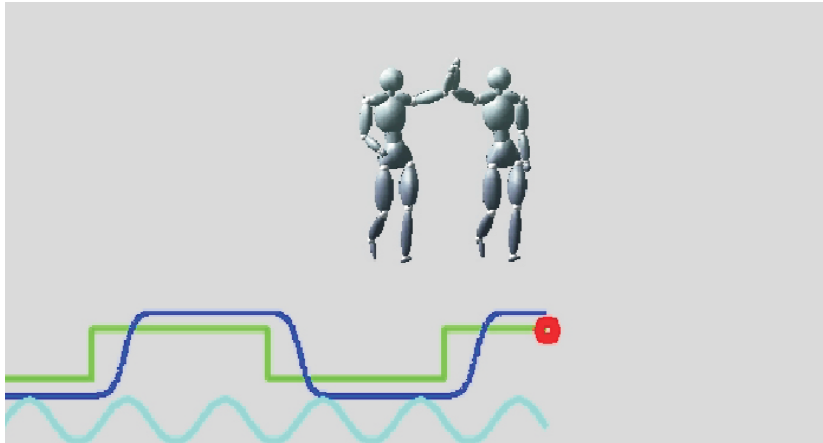
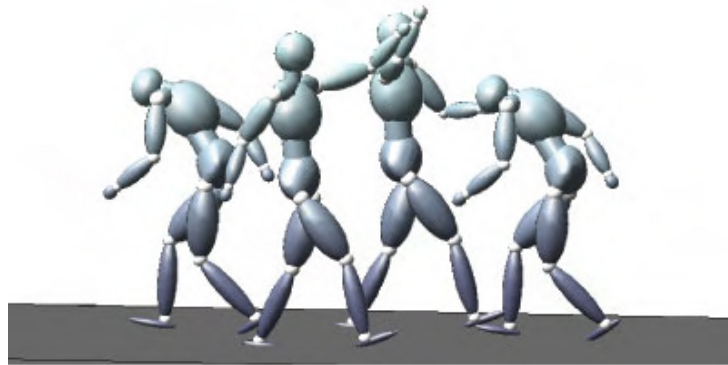


Figure 5.6: Simulation of an arm up movement by recombining the learned synergies with dynamically changing morphing weights. Cyan line: Periodic movements of the feet. Blue line: Fourth non-periodic source. Green line: External trigger signal

presented in the Figure 5.6. In this case, an arm movement is superposed with different relative timings to the periodic movements of the feet of two avatars. The sequence is generated by using the blending method described in Section 5.1.1 and by recombining the learned synergies with dynamically changing morphing weights. To initiate the raising and lowering of the arms of the avatars, an external trigger signal was included.

The same method can be applied for more complex scenarios, like two dancing couples. As illustrated in Figure 5.7, one of the two couples forms a bridge with the arms while moving forward. In the meantime, the second couple walks through this bridge one-by-one in a crouched posture. Then the partners turn around and change roles. The whole scenario was simulated online, modulating the dynamics by few binary control signals that define the action mode of each avatar (forming bridge, crouching, or turning). In this case, periodic and non-periodic movement primitives were coupled in a way that permits an initiation of the arm movement at any time during the step cycle (e.g. depending on whether the partner has already completed his turning step).

a)



b)



Figure 5.7: Dancing figures from a folk dance. The sequence was generated on-line by blending and recombination of the learned synergies with dynamically changing morphing weights in matlab a) as well as with an animation software b). The corresponding movie can be found under **Movie** (See text for further details.)

5.1.1.2 Modulation of Walking Speed

Instead of capturing gaits with different walking speed, we have the advantage that our periodic source signals are encoded by oscillators. Different propagation speeds can be achieved by changing the eigenfrequency of the oscillators, which results in motion primitives with corresponding frequencies. For modifying the gait speed during walking, linear blending can be used again. The adjustment is done by interpolating between different eigenfrequencies of the oscillators. It can be formalized in this equation:

$$\omega_0(t) = \eta(t) \omega_0^a + (1 - \eta(t)) \omega_0^b \quad (5.5)$$

The change of the morphing parameter can also be made dependent on the behavior of other avatars in the scene to simulate interactive behavior. For instance, the introduction of a distance control, which manages the distances between one character to another, or even force a fixed distance can be simulated. The distance-frequency coupling was implemented by a sigmoidal function:

$$\eta(t) = \xi(d(t)) = \frac{1}{1 + \exp(-\gamma d(t))} \quad (5.6)$$

with the positive constant γ . In this context, the morphing weight can be defined as a nonlinear function of the distance d from another character, as our distance-dependent control for the modulation of the walking speed: $\eta(t) = \xi(d(t))$.

The '*father-son example*' demonstrates this behavior: The son is following the father with a slower walking speed. If the distance exceeds a certain threshold between the two, the son accelerates by changing his eigenfrequency to catch up with his father, who takes the role of the leader. This behavior was simulated by making the eigenfrequencies of the oscillators of the follower dependent on the distance to the leader and is shown in Figure 5.8.

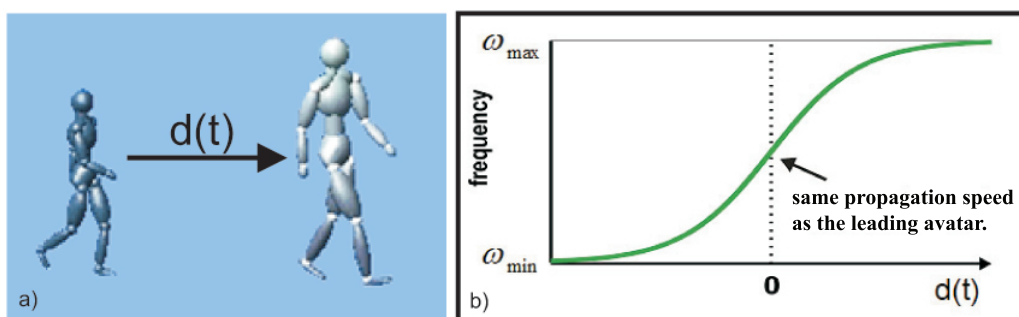


Figure 5.8: Following behavior realized by two coupled avatars. a) With the distance exceeding a certain threshold the smaller avatar accelerates to catch up with the leader b) Eigenfrequency ω_0 dependent on the distance $d(t)$ between different characters. [cf. **Movie**]

5.1.2 Character Propagation

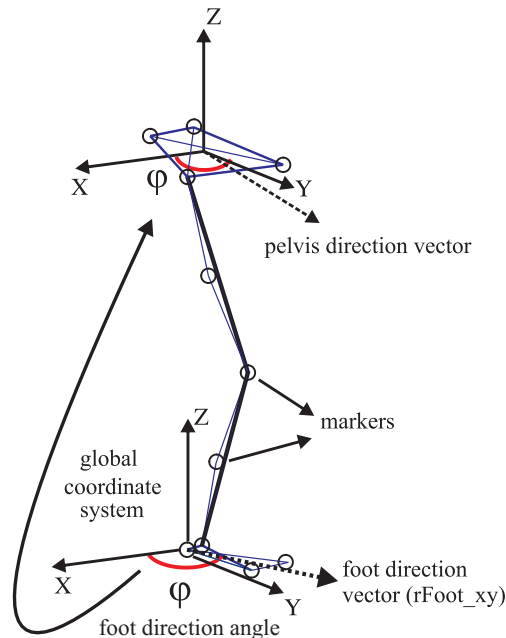


Figure 5.9: Pelvis scheme: In every time-step we compute the differential changes in the pelvis xy -direction angle in respect to the foot-on-the-ground xy -direction angle.

Walking is a natural process for the human body, but a complex one nevertheless. Not only the feet have to move across the ground, but other body parts –the head, shoulders, arms, spine, and hips– are moving in synchrony to keep the system in balance. During the walking cyclic pattern, body movements are repeated step by step, whereby the locomotion is a procedure where the moving body is supported by alternating feet: First one leg and then the other leg swing forward in preparation for its next support phase. Therefore, at least one foot always rest on the ground. During that period when the support of the body is transferred from the trailing leg to the leading leg, there is a brief period when both feet are on the ground. As a person walks faster, these periods of double support become smaller fractions of the walking cycle until, as a person starts to run, they disappear altogether, and are replaced by brief periods when neither foot is on the ground. The cyclic alternation of the support function of each leg and the existence of a transfer period when both feet are on the ground are essential features of

the locomotive process. As the body passes over the supporting leg, it rises until the foot is directly underneath and then descends again as the foot passes behind. The highest point of elevation occurs when the speed is the lowest, and the lowest point of elevation occurs when the speed is the highest. However, after the foot touches the ground, the body's weight is transferred to it and these displacements of the entire body through space can be described as a translation [RG06]. The entire body must be moved across the ground at the exact same rate that the feet are moving to maintain the illusion of walking in our character animation, otherwise, the character's feet appear to slip.

To detect the footplant in real-time for computing the displacement velocity, we propose an adaptive on-line method which takes into account the foot-floor contact constraints. Thereby the pelvis forms the root of the kinematic chain of our avatar model. We established an algorithm for computing the propagation, determining horizontal pelvis translation and the rotation of the avatar in a way that prevents foot skating.

For this purpose, we first automatically detect whether the feet make ground contact using a simple threshold criterion, for the vertical position z of the foot centers. This criterion, however, works well for the original motion capture data, assuming that the motions itself do not contain foot slipping. Since the data is not always free from noise and artifacts, we additionally add different constraints to obtain the right translation. Thereby, the correction is based on the foot with the lowest vertical coordinate z , as well as it is based on the supporting foot that touches the ground ahead of the pelvis. In this way, we then can obtain the translation rate $translation_{xy}$ from this foot considering the xy coordinates of the heel and the toe, which present in parallel the directionality of the feet and therefore also the rotation $rotation_{xy}$ for the body. The computation of translation and rotation (Figure 5.9) from foot-ground contact can be illustrated as follows:

```

for  $i \leq maxTime - 1$  do
  if  $lFoot_z \geq rFoot_z$  then
     $translation_{xy} = rFoot_{xy}(i + 1) - rFoot_{xy}(i)$ 
     $rotation_{xy} = atan2(rFoot_x(i+1), rFoot_y(i+1)) - atan2(rFoot_x(i), rFoot_y(i))$ 
  else
    if  $rFoot_z \geq lFoot_z$  then
       $translation_{xy} = lFoot_{xy}(i + 1) - lFoot_{xy}(i)$ 
       $rotation_{xy} = atan2(lFoot_x(i+1), lFoot_y(i+1)) - atan2(lFoot_x(i), lFoot_y(i))$ 
    end if
  end if

```

end for

We now add the computed differential translation and horizontal rotation to the pelvis coordinate system to animate a natural looking locomotion with minimal foot skating. To achieve an even higher degree of realism, we shift the pelvis also in the vertical axis corresponding to the heel z -translation in order to animate the natural up and down movement during walking.

5.2 Crowd Animation

The animation of crowds is a challenging task. For obtaining a realistic looking crowd behavior in real-time, each character often exhibits enormous complexity and subtlety. A crowd model must not only include individual human motion, but also address the consideration of environmental constraints such as boundaries.

This chapter involves a simple navigation model to obtain dynamic interactions between the agents. In this way, the control algorithm for the walking direction is defined by three main elements: Steering toward goals, avoiding instantaneous obstacles and avoiding predictive obstacles. Besides, we demonstrate in the second part of this chapter how interactions between multiple characters can be self-organized by implementing specific couplings within a group of, for example, pedestrians. As a result, we obtain collective and coordinated crowd animations based on the application of contraction theory (see Section 2.3), which allows a synchronized behavior in networks of coupled avatars expressed by dynamic primitives.

5.2.1 Navigation

Navigation is an essential topic in crowd animation. Humans constantly adjust their paths to navigate through a complex environment with boundaries and other dynamic factors (Figure 5.10). Even dense crowds are characterized by surprisingly few collisions or sudden changes in individual motion [TCP06]. Although the locomotion through a complex environment seems to be effortless for humans, navigation comprises different challenging tasks, such as steering towards goals or avoiding static and dynamic obstacles, e.g. other pedestrians. In this context, our model has to be extended by including realistic looking navigation dynamics to adopt the virtual characters into the surrounding and let them propagate in the virtual scene. Therefore, we invented navigation policies based on three control dynamics, which describe:

1. Moving towards a goal.
2. Avoiding stationary and dynamic obstacles.

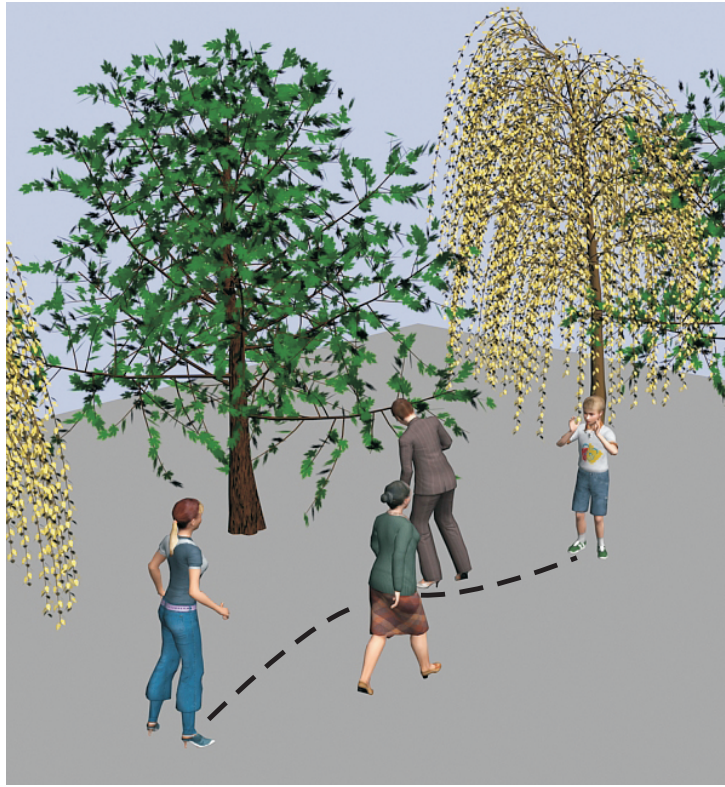


Figure 5.10: Humans constantly adjust their paths to navigate through a complex environment.

Our dynamic navigation model is originally inspired by robotics. We combined our trajectory generation algorithm with a simplified version of a dynamic navigation model that has been successfully applied before in robotics [SDE95, War06].

In this context, goals can be described by attractors or regions in state space towards trajectories converge. Conversely, states to be avoided, like obstacles behave as repellers –regions from which trajectories diverge– and force back the heading direction of the agent. Thereby, the strengths of the repeller depend on the distance between the obstacle, and the character, which decreases with increasing distance. The locomotion behavior was modeled by a differential equation in which the angular acceleration is described as a function of the goal, the obstacle angle, and the distance.

We extended this model by including a predicted obstacle term, which influences the behavior of the heading course by detecting a possible collision. Hence, the navigation dynamics were given by a differential equation for the heading directions φ_i of the characters and can be characterized by linearly combining the terms

for the goal, the obstacle, the predictive obstacle, and the distance. The turning rate of the avatars was controlled by morphing between straight and curved walking steps (Section 5.2.1). Whereby, the morphing weights were dependent on the temporal change of heading direction $\dot{\varphi}_i$.

5.2.1.1 Navigation Algorithm Outline

The navigation dynamics specifies the direction change $\dot{\varphi}_i$ of the character by a differential equation that integrates three different components, where \mathbf{p}_i denotes the position of character i :

$$\begin{aligned} \frac{d\varphi_i}{dt} = & \underbrace{h^{\text{goal}}(\varphi_i, \mathbf{p}_i, \mathbf{p}_i^{\text{goal}})}_{\text{goal-finding}} + \underbrace{\sum_j h^{\text{avoid}}(\varphi_i, \mathbf{p}_i, \mathbf{p}_j)}_{\text{instantaneous obstacle avoidance}} \\ & + \underbrace{\sum_j h^{\text{pcoll}}(\varphi_i, \varphi_j, \mathbf{p}_i, \mathbf{p}_j)}_{\text{predictive obstacle avoidance}} \end{aligned} \quad (5.7)$$

The two-dimensional vectors \mathbf{p}_i and \mathbf{p}_j signify the 2D-positions of the character i and the moving obstacle j - in our case another pedestrian- in the scene, and the variables $\varphi_i(t)$ and $\varphi_j(t)$ their heading directions, which can be estimated from the momentary velocities.

Goal-Finding Term: The first term in the vector field permits to define a goal position $\mathbf{p}_i^{\text{goal}}$ that the character tries to approach. We can represent this by a term that specifies the turning rate for each possible value of heading. If the heading increases away from the goal, the dynamics introduces a force that steers the avatars towards the goal, e.g. Figure 5.11 a). This can be expressed by the following term:

$$h^{\text{goal}}(\varphi_i, \mathbf{p}_i, \mathbf{p}_i^{\text{goal}}) = -\sin(\varphi_i - \varphi_i^{\text{goal}}) \quad (5.8)$$

where φ_{goal}^i is the direction angle of the goal relative to the position \mathbf{p}_i in external coordinates.

Stationary and Dynamic Obstacle Avoidance: The second term implements obstacle avoidance, where the obstacles \mathbf{p}_j define stationary, as well as dynamic objects like other avatars (Figure 5.11 b). Therefore, the dynamics of obstacle avoidance correspond to repellers that push the character's heading away. This repelling behavior of the obstacle decreases with distance and can be represent by

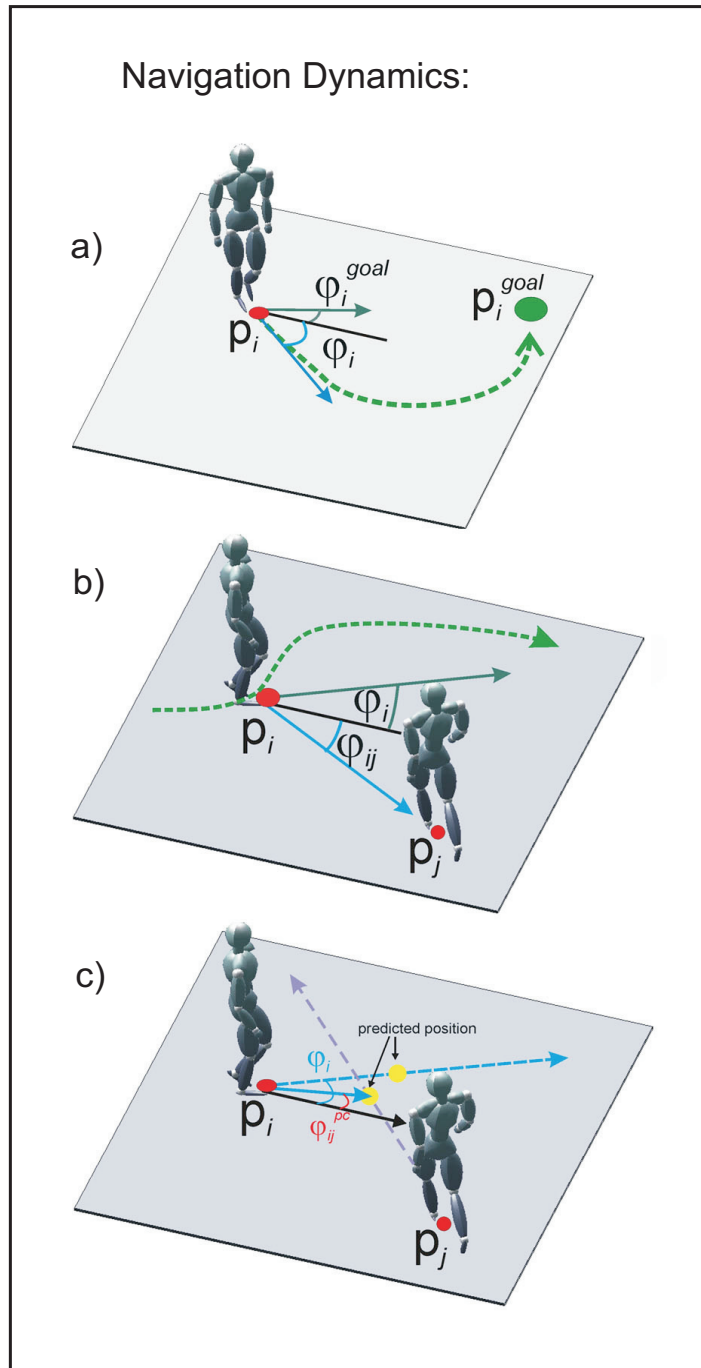


Figure 5.11: Navigation dynamics depending on: a) Goal-finding term, b) instantaneous obstacle-avoidance term, and c) predictive obstacle-avoidance term.

multiplying the obstacle angle $\varphi_i - \varphi_{ij}$ by an exponent of minus squared distance. This term is given by the following expression:

$$h^{\text{avoid}}(\varphi_i, \mathbf{p}_i, \mathbf{p}_j) = \sin(\varphi_i - \varphi_{ij}) \cdot \exp\left(-\frac{(\varphi_{ij} - \varphi_i)^2}{2\sigma_\varphi^2}\right) \cdot \exp\left(-\frac{d_{ij}^2}{2\sigma_d^2}\right) \quad (5.9)$$

where $\varphi_i - \varphi_{ij}$ defines the relative direction of character j from character i in global coordinates. Introducing the vector $\mathbf{d}_{ij} = \mathbf{p}_j - \mathbf{p}_i$ one obtains

$$\varphi_{ij} = \arctan((\mathbf{d}_{ij})_y / (\mathbf{d}_{ij})_x) \quad (5.10)$$

and $d_{ij} = |\mathbf{d}_{ij}|$, the 2D distance between the two characters. The values of the constants were $\sigma_\varphi = \pi/6$ and $\sigma_d = 1.5 \dots 2$ m.

In order to remove far field interactions, we set h^{avoid} to zero for $|\Delta\varphi_{ij}| \geq \pi/2$ and for $d_{ij} > 4$ m. Headings to the right of the obstacle yield a positive turning rate that asymptotes to zero as the agent turns away from the obstacle, and headings to the left of the obstacle yield a negative turning rate that also asymptotes to zero. In this way, the heading direction is forced away from the dynamic or stationary obstacles.

Predictive Obstacle Avoidance: A much more realistic collision avoidance is accomplished by the inclusion of a third term in the navigation dynamics that is dependent on the predicted future positions of the avatars (Figure 5.11c). When a collision is likely to occur in the future, this third term helps to prevent collisions by steering the characters away from each other at an early stage. The prediction assumes straight trajectories of the avatars and computes the closest point between their predicted trajectories. The implemented force depends on the future positions of the characters i and j , computed from their current positions and momentary velocities $\mathbf{v}_i, \mathbf{v}_j$. We define $\varphi_i - \varphi_{ij}^{\text{pc}}$ as the direction to the character j at the expected moment of minimal distance. With $\mathbf{v}_{ij} = \mathbf{v}_j - \mathbf{v}_i$ and $\mathbf{d}_{ij} = \mathbf{p}_j - \mathbf{p}_i$ the characters approach each other only for $\mathbf{v}_{ij}^T \mathbf{d}_{ij} < 0$. In this case, the closest relative position in the future occurs after the time $\tau^{\text{pc}} = \mathbf{v}_{ij}^T \mathbf{d}_{ij} / (\mathbf{v}_{ij}^T \mathbf{v}_{ij})$. This third term is defined by:

$$h^{\text{pcoll}}(\varphi_i, \varphi_j, \mathbf{p}_i, \mathbf{p}_j) = \sin(\varphi_i - \varphi_{ij}^{\text{pc}}) \cdot \exp\left(-\frac{(\varphi_{ij}^{\text{pc}} - \varphi_i)^2}{2\sigma_\varphi^2}\right) \cdot \exp\left(-\frac{(d_{ij}^{\text{pc}})^2}{2\sigma_d^2}\right) \quad (5.11)$$

where the constants σ_φ and σ_d were chosen as before and h^{pcoll} was set to zero for $\varphi_i - \varphi_{ij}^{\text{pc}}$ and $d_{ij}^{\text{pc}} > 8$ m.

5.2.1.2 Control of Walking Direction

As we have seen in Section 5.1.1, different motion styles were synthesized by applying morphing methods, which are also used in a similar way to implement direction changes of the avatars for navigation purposes. The change of the morphing parameter η can be made dependent on the behavior of other avatars in the scene, e.g. to influence the emotional style depending on the distance of the avatar from another 'dangerous' colleague. Hence, the walking direction of the characters is computed by interpolation between straight walking and walking along curved paths to the left (for $d\varphi_i/dt > 0$ or to the right ($d\varphi_i/dt \leq 0$) using equation (5.4). The morph parameter was taken proportional to $|d\varphi_i/dt|$, normalizing it in a way that ensures $\eta = 1$ for the maximum possible value of this derivative. The heading direction $\varphi_i(t)$, generated by the navigation dynamics, was low-pass filtered with a time constant equal to one step cycle as to improve the smoothness of the navigation behavior.

Navigation with Emotional Changes

As one example is illustrated in Figure 5.12, a group of avatars that meets in the center of the scene changes their affect upon the contact with the other characters. This behavior was implemented by making the affect of each avatar dependent on the distance from the others. In addition, the avatars avoid each other due to the navigation dynamics described in Section 5.2.1. In this simulation, navigation and changes of emotional styles were combined, based on only three prototypical gaits: neutral walking with rotation right or left and emotional straight walking.

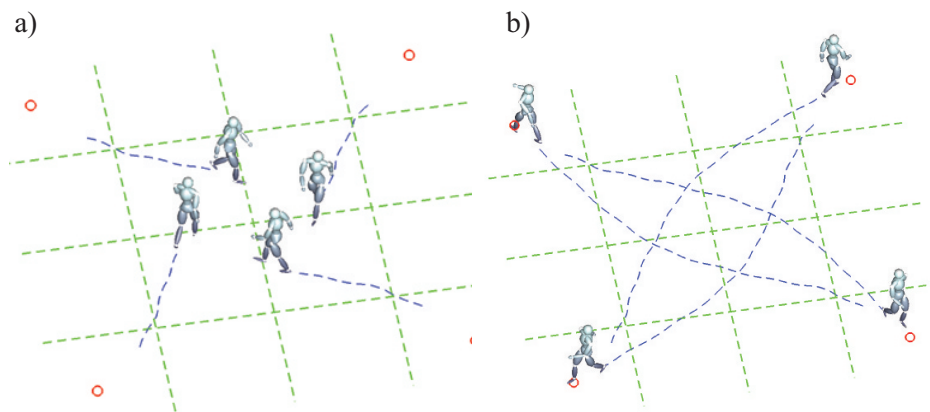


Figure 5.12: Avoidance behavior and change of emotional style. a) Avatars starting from different positions with a sad emotion are heading towards their goals (red circles). b) At the meeting point the emotional styles change to happy. In addition, the characters avoid each other. **DEMO**

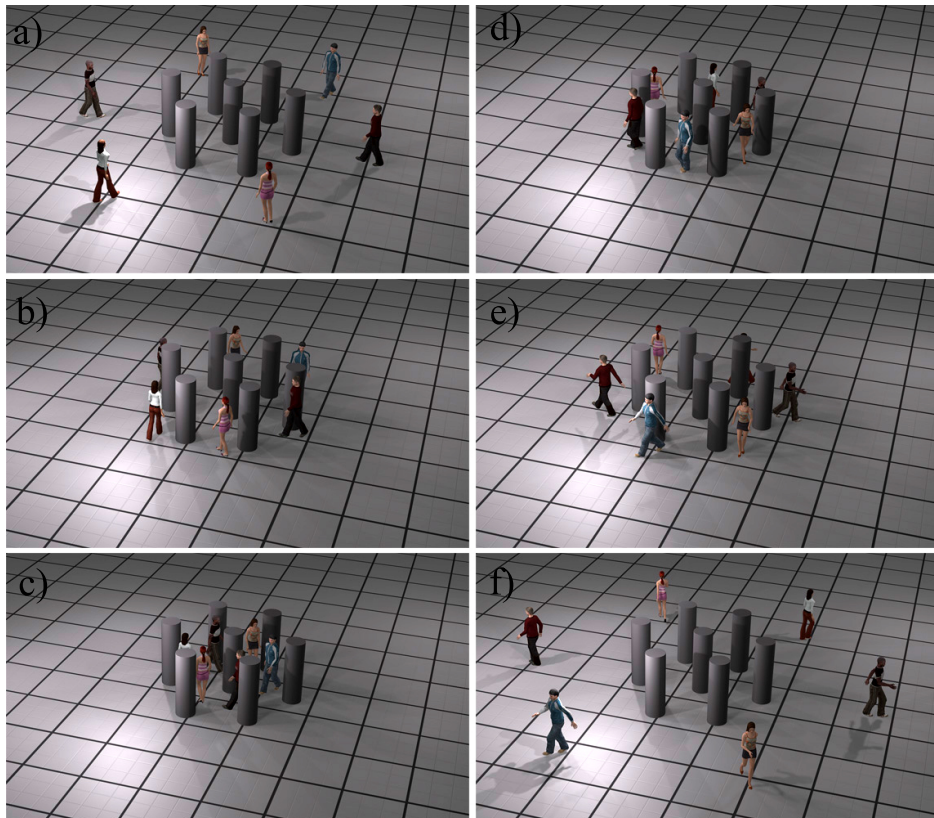


Figure 5.13: Simulation of stationary and dynamic obstacle avoidance, while changing the emotional styles and speed interactively. a) The avatars' goal areas are positioned on the opposite side of the scene's floor. b), c), d) Static obstacles (columns) and dynamic obstacles (other avatars) have to be avoided simultaneously. e), f) After avoiding all obstacles, the characters proceed heading towards their goal areas. The demo, corresponding to the described scene can be seen in **DEMO**.

The navigation dynamics not only enable an interactive behavior within the crowd, but also with the dynamic environment. Stationary obstacles, or other crowd members, can be avoided autonomously without planning the path for each agent separately (Figure 5.13). Moreover, we exploit this framework by going a step further and combine the navigation dynamics with emotional styles to make it even more interactive.

In order to produce the morphs between straight, emotional gaits and neutral curved walking (left and right), we first created an intermediate balanced mixture by interpolating the mixing weights according to the relationship:

$$w_{ij} = \frac{3}{4}w_{ij}^{\text{emotional}} + \frac{1}{8} \left((1 + \beta_{LR})w_{ij}^{\text{Left}} + (1 - \beta_{LR})w_{ij}^{\text{Right}} \right) \quad (5.12)$$

The parameter β_{LR} , with $0 < \beta_{LR} < 1$, was adjusted for different emotional styles in order to balance left-right declinations from the straight line. In accordance with Section 5.2.1, morphing was done in a piece-wise linear manner dependent on the sign of the change of the heading direction.

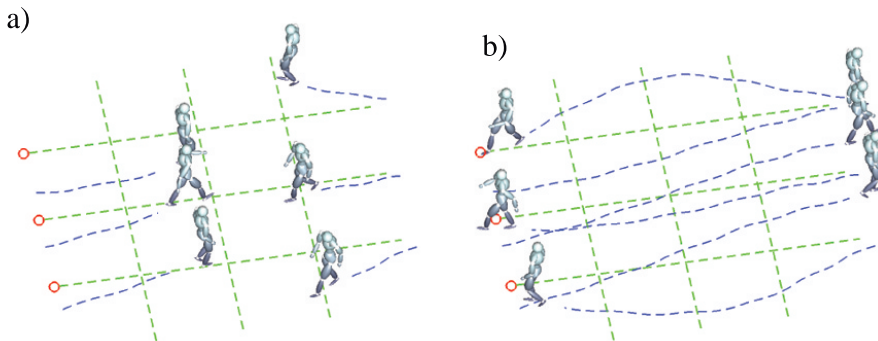


Figure 5.14: Avoidance behavior and change of emotional style. Three avatars, starting from the left side, change their emotion from happy to sad while proceeding to their goals. A second group of avatars starting from the right side change their emotions from sad to happy while avoiding the opposing group. **DEMO**

Another simulation based on a similar implementation is shown in Figure 5.14. In this particular situation, three avatars, starting from the left side, change their emotion from happy to sad while proceeding towards their goals. A second group of avatars, starting from the right side, change their emotions from sad to happy while avoiding the first group.

This behavior was implemented by designing dynamics that makes the emotional style of each avatar dependent on the distance from the others. In addition, we ensure that the avatars approach their goal points while avoiding each other. Again, this simulation combines navigation and changes of emotional style dependent on the distance to the counter group. It is based on only three prototypical gaits that were recorded by motion capture: neutral walking with rotation right or left, and two emotional walking styles.

5.2.2 Self-Organized Behavior in Crowds

The magnificently synchronized coordination of collective motions in flocks (e.g. fish swarms) are characterized by their visual complexity despite arising from simple rules that result from the dynamic interactions among individual agents, and without central coordination. By observing such biological organizations, scientists try to deduce new coordination approaches as in autonomous dynamics. Especially in robotics, the principle of cooperative control is used to enable a large group of autonomously functioning vehicles in the air, on land or in the sea, to collectively accomplish useful tasks in a coordinated manner (e.g.[[Cou09](#)]).

Furthermore, the application of group coordination or collective behaviors is a key subject in computer graphics for crowd animation. We describe a crowd by a group of individuals in the same environment sharing, a common goal. The motion of an agent in a crowd is often computed separately, where the scene is manually composed using singly captured motions or keyframe animations [[SKSY08](#)]. Although the simulation of close interaction can be fulfilled and each individual can be treated on his own (e.g. making their own decisions), it requires much computational resources and much expertise and labor by the animator (Section [1.3.1.1](#)). In addition, it is very difficult to develop behavioral rules that consistently produce realistic motion. Global path planning, especially in real-time becomes computationally expensive, particularly regarding every single character within the crowd.

Hence, it is not surprising that the generation of autonomous behaviors for many characters require complex architectures of dynamical systems. Nevertheless, we managed to simplify the architecture by presenting realistic human movements with very few simple dynamical systems. Introducing couplings between agents enable coordination to simulate collective behavior. Thereby, varying synchronized and completely self-organized crowd scenarios have been obtained by applying only different coupling structures (see Chapter [6](#) for more details). In this way, we accomplish a synchronized crowd behavior through a cooperative control model.

5.2.2.1 Coordinated Crowd Behavior

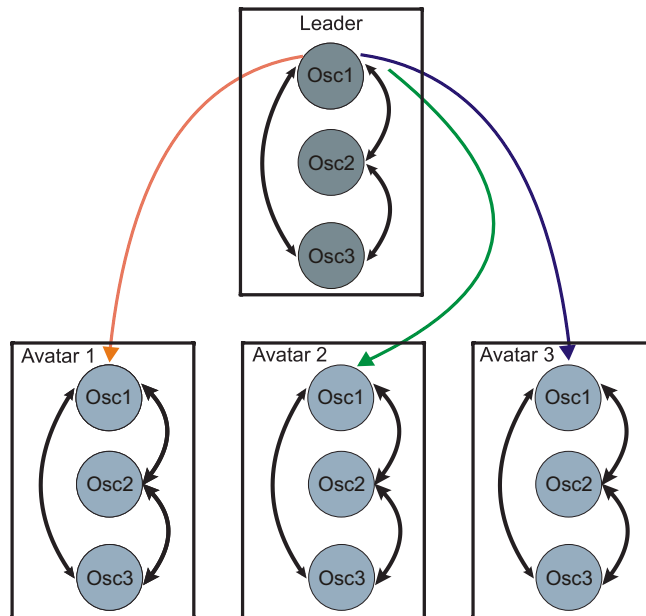


Figure 5.15: Coupling of multiple avatars, each of them comprising three coupled oscillators (Osc1, Osc2, Osc3), permits the simulation of the behavior of coordinated crowds.

The variety of periodic movements has been reduced into very few components together with style dependent weights and delays. The desired motion can be reconstructed for real-time animation, by using nonlinear dynamical systems which are associated with the movement components applying unsupervised learning (Chapter 4). Accordingly, each character is expressed by a network of dynamic primitives or more specifically: limit-cycle oscillators. To accomplish a more robust behavior against e.g. disturbances, we introduced velocity couplings between the oscillators, thus obtaining a more coordinated motion (Section 4.3.4).

The stability analysis based on contraction theory see Chapters (4,6) permits the design of such dynamical systems and promises a stable overall system architecture. Being aware of these particular coupling techniques, we are able to extend this framework of coupled oscillators and design a network within many interconnected characters or subnetworks. Therefore, the same type of coupling method, as in Section 4.3.4, has been applied resulting in a synchronisation behavior in crowds. By introducing unidirectionally couplings, it is possible to make multiple characters following another one automatically, who then acts as a leader (Figure

5.15). Therefore, we only connected the oscillators with each other, which corresponds to the source with the lowest frequency (**Osc1**). As a result we are able to self-organize coordinated behavior of crowds with relatively realistic appearance by applying simple coupling techniques. This can be illustrated schematically in Figure 5.16, which shows a few snapshots from an animation where a group of characters begins with asynchronous step phases. One of the characters acts as a guide, and the dynamical systems of the other characters are coupled unidirectionally with the leaders' dynamics (cf. Section 2.4.3).

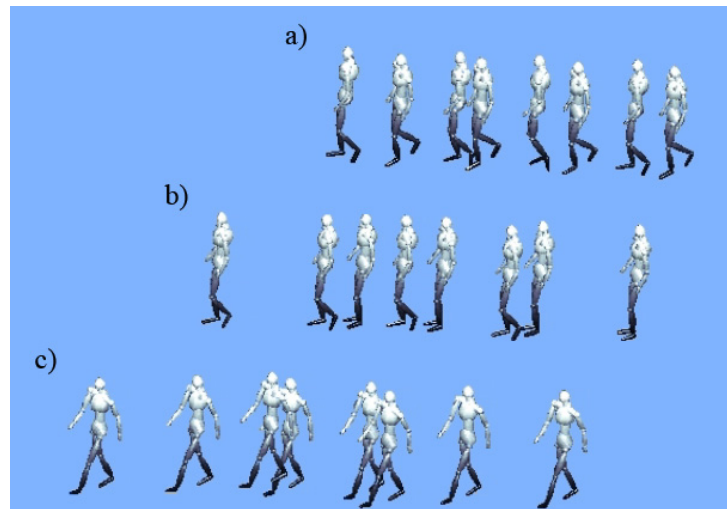


Figure 5.16: Autonomous synchronization of gait patterns within a crowd. (a) Avatars start with self-paced walking and are out of phase. After a transitory period (b), the gaits of the characters become completely synchronized (c).

After a short transition period the step cycles of all characters synchronize with the leader.

This has the consequence of the crowd leaving the scene with synchronous step pattern ('lock-step'), whereby the information of the rhythm was given from only one character's dynamics. This shows that the proposed method is suitable for the simulation of coordinated behavior of crowds, like marching soldiers. Another example consists of several dancing scenes that require the coordination of locomotion patterns. A demonstration video can be seen here **DEMO**.

Synchronized Behavior in Comparison

In the context of the scenario above, we compared our method with standard approaches such as PCA (cf. Figure 5.17 with corresponding demo. Avatar 3 acts as a 'leader' because it is driven by three coupled oscillators, but without additional external couplings. The other two avatars are coupled to this leader and

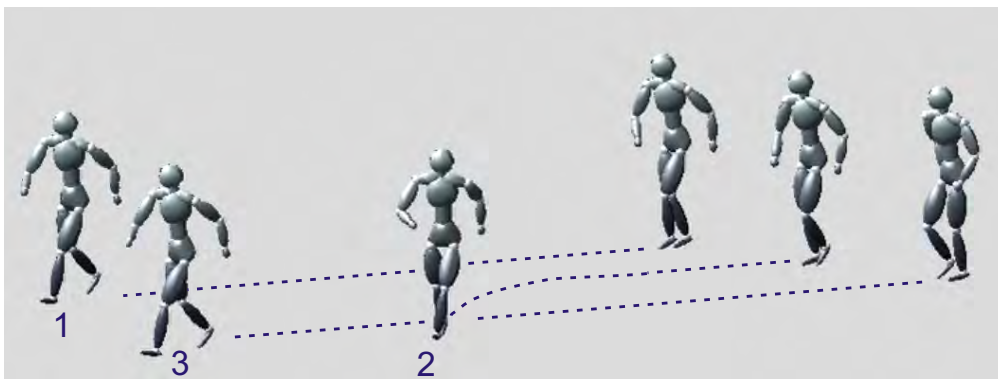


Figure 5.17: This example video shows the comparison of a synchronization behavior of a model, which is generated by our method with a standard approach such as PCA. See text for more details. [DEMO]

start with equal initial phases that are different from the phase of the leader. The movement of one character (avatar 1) was generated applying our novel method, using an anechoic mixture model with three sources. The movement of the second character (avatar 2) was generated with a PCA model with seven components in order to obtain the same approximation quality. This avatar was driven by seven coupled Van der Pol oscillators where we attempted to optimize the coupling for maximum naturalness of the obtained animation.

The detailed comparison shows that the avatar whose motion was generated by the novel architecture (oscillator dynamics with 6 degrees of freedom) shows a quite natural-looking transition from its initial state to the equilibrium state that is synchronized with the leader. The movement of the avatar whose movement was generated using PCA (oscillator dynamics with 14 degrees of freedom) shows artifacts. The amount of artifacts is even increased if transitory body motion is added by enforcing the kinematic foot-contact constraints on the ground, resulting in a turning motion of the avatar (see Figure 5.17). If the internal coupling strength within the avatars is increased, the synchronization time between multiple avatars slows down and an unnatural reduction of step size arises. If the number of components in the PCA model is increased to 12, similar problems remain also for stronger coupling forces [DEMO].

The proposed novel trajectory model thus tends to produce more natural transitions between different coordination states. Present work focuses on a more systematic quantitative comparison between different methods. However, a model consisting of three oscillators synchronizes faster and produces more natural-looking behavior.

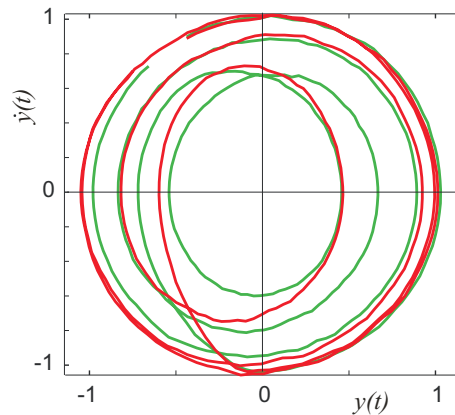


Figure 5.18: Phase plot of the first (principal) oscillators of two avatars that are coupled to a leader. The diagram shows the state space of the oscillator in the $[y(t), \dot{y}(t)]$ plane. The green line corresponds to the avatar driven by five PCA components, and the red line to the avatar animated with three sources extracted with our algorithm. The coupling is switched on after a short starting period.

The same example also demonstrates that the model, even though it has been trained in the attractor of the dynamics, generalizes in a meaningful way to transient states. This is illustrated by the phase diagram in Figure 5.18 that illustrates the trajectories of the first leading oscillators of the two avatars. In the first place, this figure shows that the oscillators' states are not near the attractor state during the synchronization period, requiring the system to generalize to trajectories that were not used during the training of the system. In the second place, it is obvious that the system with less oscillators (red trajectory) returns faster to the attractor than the system with five oscillators (green trajectory). More detailed investigations testing dependence on coupling strength and coupling structure are underway. Since the linear weights of the PCA components are not sparse, the mixing of the many oscillator inputs results in jerky motion in the starting phase of the synchronization period.

Self-Organized Crowd-Scenario

The methods mentioned before –morphing, navigation and introducing couplings– can be combined to explore the capabilities of the proposed framework: A completely self-organized scenario of a larger group has been simulated for a dance scenario, where the avatars have to walk in synchrony with the music in formation. In this sequence, eight pairs of avatars execute a type of 'folk dance' that requires them to walk along a straight line, forming a corridor. Once the couples have reached the end of the corridor, they have to walk quickly to the other end and re-enter the corridor from the other side. At the point of re-entrance, the individ-

ual avatars need to synchronize with their partner. Despite this relatively complex scenario, the whole group of 16 avatars can be self-organized quite naturally.

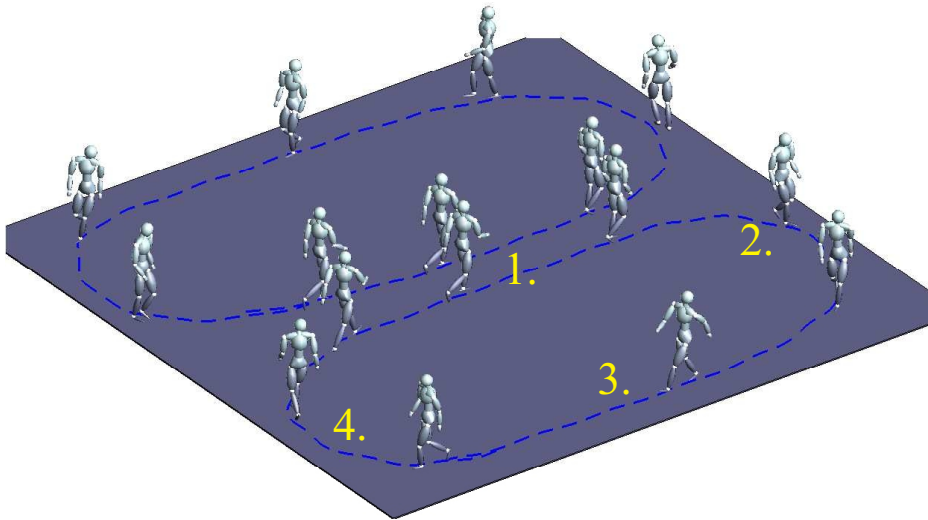


Figure 5.19: Simulation of a 'folk dance' behavior is organized within four zones: 1) Avatars within the corridor move in synchrony, simulated by coupling the corresponding oscillators. 2) Leaving the corridor, the avatars of the individual dancing couple become separated and start to move along curved paths. Their emotional style changes from happy to neutral. 3) Avatars walk asynchronously and 'hurry up' to reach the entrance of the corridor, simulated by increasing the eigenfrequency of the oscillators. 4) When the avatars approach the entrance of the corridor their walking speed decreases, and they need to get in synchrony again with the appropriate leg. This can be simulated by appropriate adjustment of the oscillator frequencies. [DEMO]

To simulate this complex behavior, we divided this scenario into four main sectors that are described separately in the following (see Figure 5.19):

1. At the entrance of the corridor, the characters gather and wait for the corresponding partner to move synchronously. To simulate the synchronized walking of all couples within the corridor the corresponding oscillators are coupled to equation (4.10), introducing couplings not only between the avatars of one couple but also between the subsequent couples within the corridor and a coupling to an external periodic signal derived from the music.

2. At the end of the corridor, reaching the second zone, the two avatars of each couple separate and the coupling between their oscillators is removed. This results in an asynchronous movement that is controlled by the navigation dynamics (Section 5.2.1). In addition, within this zone the emotional walking style of the characters changes from happy to neutral. The curved walking paths were generated by defining appropriate intermediate goal points.
3. Along the straight paths outside the corridor, the avatars accelerate to catch up with their associated partner in time at the beginning of the corridor, which is simulated by a temporary increase of the eigenfrequency of the corresponding oscillators.
4. In the last zone, the characters decelerate, modeled by decreasing the frequency of the oscillators. A difficult problem is the re-synchronization with the correct foot at the entrance of the corridor. This is accomplished by slightly adjusting the oscillator frequencies to ensure re-synchronization with the appropriate leg. Again, the curved paths are generated by defining appropriate goal point exploiting the navigation dynamics (Section 5.2.1).

Another visualization of a completely self-organized example, is illustrated in Figure 5.2.2.1 with the corresponding movies. They describe a formation scenario of 'soldiers': First, a group of characters are placed randomly in the scene without including any couplings. To obtain the desired formation, we included a distance-frequency coupling (Section 5.1.1.2) among them to ensure collocation and keep the others at the right distance. Because in such a soldier formation, each individual has to march in lock-step, we additionally determine a distance-to-step size coupling. As a result the soldiers not only get ordered, but farther they start to synchronize with each other by adapting the step size. Meanwhile each soldier is steering towards to his own predefined goal (2.3 meters ahead of him) and avoiding the other members within the group using navigation dynamics (Section 5.2.1).

In this manner, we are able to create diverse scenarios of interactive autonomous behavior in crowd animations. From a small database of motion clips it is possible to obtain a huge variety of different motion styles, based on a simple dynamic architecture, which give us the opportunity to simulate large self-organized groups in an efficient way. Nevertheless, the control of human locomotion is a complex task –comprising nonlinear dynamics– that have to consider the coordination for collective crowd behavior and the navigation through the changing environment. In order to investigate stable conditions for such simulated crowd scenarios, the stability analysis is needed to ensure a stable over all architecture.



Figure 5.20: Self-organized formation behavior of interacting characters ('soldiers') in crowds. Different coupling methods and navigation dynamics have been applied to observe the coordinated behavior. [DEMO1] or [DEMO2].

Stability Properties in Character Animation



Figure 6.1: Illustration of a large crowd animation in video games (Napoleon: Total War).

The magnificently synchronized coordination of the collective motions in flocks (e.g. fish schools) is characterized by their visual complexity, but arises from simple rules and no central coordination. By comprehending such biological organizations, scientists attempt to analyze this phenomenon to learn more about coordinated behavior. In robotics, those interested in group coordination and cooperative control use this principle to enable large groups of autonomous vehicles by land, sea or air, to collectively accomplish useful tasks, in a coordinated man-

ner (e.g. [Cou09]).

Especially for large crowd scenarios, the understanding of such coordinated organizations would help to simplify the complexity in crowd animation:

In this context, the critical limiting factor is the complexity of the dynamical models for individual agents or characters, which often make a systematic treatment of stability properties infeasible. Therefore, instead of deriving the system parameters from theoretical results on the system dynamics, the design of such systems has been often heuristic, meaning that empirical results from simulations are exploited. However, the controlled engineering of such system makes more systematic theoretically founded approaches highly desirable.

In Chapter 4, a method was developed that approximates complex human behavior by relatively simple nonlinear dynamical systems. Consistent with related approaches in robotics [GRIL08, AMS97, SIB03] and biology [FH05], this method generates complex movements by combining learned movement primitives.

The resulting system architecture is rather simple and thus suitable for a treatment of dynamical stability properties. While the design of stability properties is a central topic in robotics [BRI06, RI06, GRIL08], it is rarely addressed in character animation.

Nevertheless, the simulation of collective behavior by self-organization in systems of dynamically coupled agents seems interesting for several reasons:

First, it might help to reduce the computational costs of traditional computer animation techniques, such as scripting or path planning [TCP06, TWP03]. In addition, the generation of collective behavior by self-organization results in spontaneous adaptation to perturbations or changes in the number of characters [CS07, OEH02].

This chapter exploits character models, whose behavior is driven by nonlinear limit cycle oscillators. The stationary solution of these oscillators is given by a sinusoidal oscillation with a constant equilibrium amplitude.

Groups of interacting characters can be modeled by coupled networks of such nonlinear dynamical systems. By applying contraction theory, stability properties for various coupling structures have been observed (Chapter 2.3) in order to design stable scenarios for interactive character animations. Coordinated group behavior of multiple characters with different coupling conditions and the corresponding behavior have been simulated and will be discussed later.

6.1 Collective Behavior Fulfilling And Violating Contraction Bounds

Physical or biomechanical models of complex human movements are typically complex and require about 30 degrees of freedom for obtaining a sufficient extend of realism [HWBO95, GTH98]. Nonlinear systems of this complexity are typically not accessible for a more detailed analysis of their dynamical stability properties.

Our animation system models the behavior of characters by learning the mappings between stable solutions of relatively simple nonlinear dynamical systems and the corresponding very few movement primitives, representing the motion trajectories of a human. By learning the appropriate simplified models, our approach results in dynamical models with a limited degree of complexity.

A single character in our animations is described by a dynamical system of the form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, t) \tag{6.1}$$

where the variable \mathbf{x} signifies the dynamical state of the character.

For a walking avatar, this dynamic could be given by a limit cycle oscillator, whose periodic solution is mapped onto the joint angles of the character. The nonlinear mapping between the dynamical state \mathbf{x} and the joint angles is learned through the use of kernel methods (see Section 4.3.3 for details). The learned nonlinear mapping is bounded and acts as a nonlinear observer of the state variable that does not modify the overall stability properties of the system, unless the joint angles are fed back into the dynamical state. The dynamics in equation 6.1 can also be interpreted as a *central pattern generator* that drives the movement of the character. More specifically, CPGs encode rhythmic trajectories as limit cycles of nonlinear dynamical systems, usually systems of coupled nonlinear oscillators and offer multiple interesting features: For instance, the stability properties of the limit cycle behavior (i.e. perturbations are quickly forgotten) or the smooth online modulation of trajectories change in the parameters of the dynamical system.

Contraction theory was applied in Section 2.3 onto networks consisting of coupled limit-cycle oscillators. Analyzing networks with different coupling structures such as all-to-all, chain or leader-group-coupling, contracting conditions, boundaries for animating synchronized scenarios and estimation of relaxation times were obtained.

The following section presents a number of examples illustrating the behavior of groups of characters when the underlying dynamics fulfill or violate the bounds for contracting system behavior, whose contracting boundaries have been com-

puted before in Section 2.4.

6.1.1 All-to-All Symmetric Coupling

The first set of demonstrations shows the synchronization within a group of three characters. The characters are, therefore, connected symmetrically with each other and were simulated with different coupling strengths k . As it was discussed in Section 2.3, for a contracting network of three symmetric coupled Andronov-Hopf oscillators, the condition for k is: $k \geq 1/3$.

Example 1: In this case, the coupling strength is defined with $k = 0.334$ and thus it fulfills the theoretical bound. As a result, the dynamics quickly converges to a stable state and is been illustrated in Figure 6.2. It shows a group of three characters starting with random initial step phases and becoming synchronized during a very short period of time. Thereby, the frequency of the characters (oscillators) start adjusting their rhythm with each other and end up in a frequency entrainment, which results in synchronous walking.

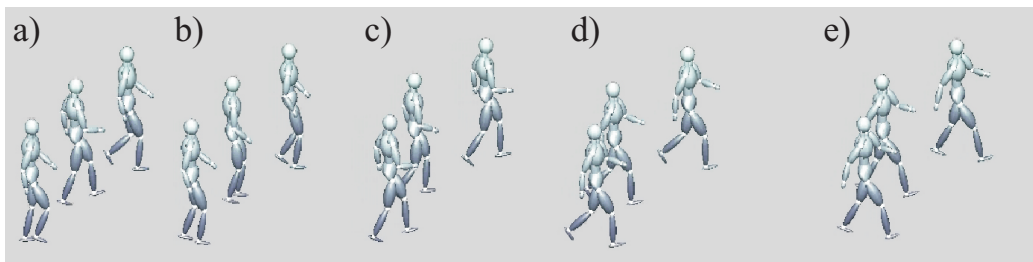


Figure 6.2: Group of characters that are all-to-all coupled and fulfill the contracting conditions ($k = 0.334$). The characters start with random initial condition (a) and adapt to the other step phases (b, c) and until they are synchronized (d, e). [Movie_1]

Example 2: Conversely, the next illustration 6.3 shows an example where the coupling strength $k = 0.111$ violates the theoretical bound. The characters start with different step-phases and do not obtain a synchronized group behavior, or at least they synchronize in a very slow manner (reaching an equilibrium state only after hundreds of time steps). The fact that the system still converges to a stable solution reflects that the bounds derived by contraction theory define sufficient but not necessary stability conditions.

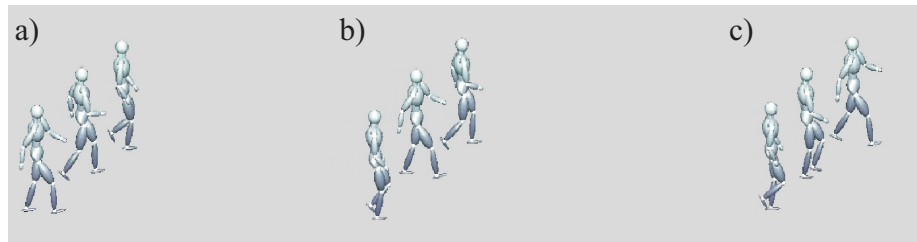


Figure 6.3: Group of characters that are all-to-all but with a violating contracting condition ($k=0.111$). The characters start with different step cycles (a) and do not synchronize (b, c). [Movie_2]

Example 3: For an even stronger violation of the theoretical bound (Figure 6.4), we choose the coupling gain of $k = -2.0$. This results in a system dynamics that does not achieve the formation of a coordinated behavioral pattern anymore. The strong coupling deforms the limit cycles in phase space, resulting in unnatural joint trajectories and a very slow propagation speed of the characters.

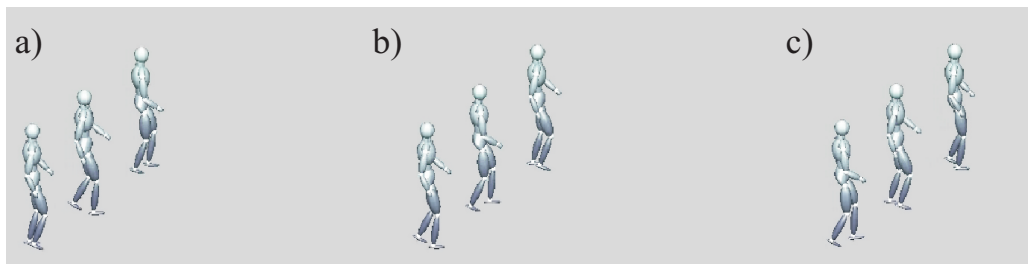


Figure 6.4: Group of characters that are all-to-all but with a violating contracting condition ($k=-2.0$). Due to the coupling gain, no coordinated behavior can be achieved and in addition, the walking movement becomes even unnatural (a-c). [Movie_3]

6.1.2 Chain Coupling

The following set of demonstrations was generated under the assumption of a bidirectional chain coupling among the oscillators. As it was evinced in Section 2.3, the bound for a contracting attitude of three coupled oscillators (characters) is given by the coupling strength of $k \geq 1$.



Figure 6.5: Group of characters with a chain structure, while fulfilling the contracting condition ($k = 1.0$): Starting with random initial conditions (a), the phase steps are adapted from each other, (b) which result in a coordinated group behavior (c). ([[Movie_4](#)])

Example 4: Figure 6.5 presents the result of the character behavior by fulfilling the theoretical bound with the coupling force $k = 1.0$. As it is also illustrated in this example, the individuals start independently with different phases and show an immediate phase-locking behavior of the oscillator, producing a synchronized behavior within the group members during a short time period. **Example 5:** We define $k = 0.333$, which violates the contraction condition (Figure 6.6). In this case, the individuals with initial conditions are not 'waiting' for each other and do not realize coordinated behavior in the observed time interval.

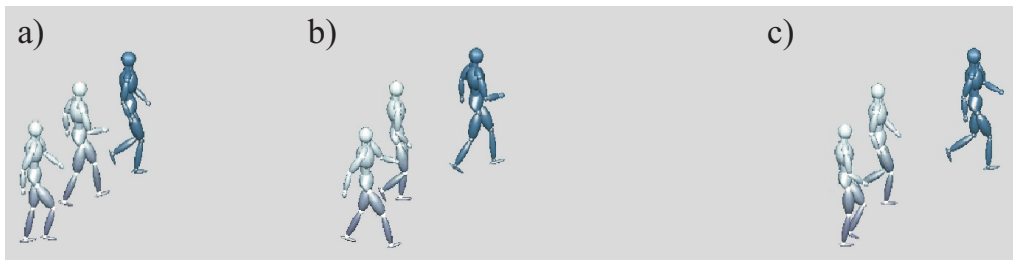


Figure 6.6: Group of characters with a chain structure but violating the contracting condition: Characters start with individual step-cycles (a) and do not synchronize (b, c). The corresponding demo can be found here: [[Movie_5](#)]

6.1.3 Leader-Group Interaction

As discussed in Section 2.4.3, one can introduce a leader that can entrain all other characters in the scene by its own behavior. In addition, coupling with a leader can synchronize other characters in the scene that would not synchronize otherwise. In Section 2.4.3 we showed that contracting behavior is obtained for $kN + \alpha > 1$,

assuming that k signifies the coupling strength between the members of the group and α the strength of the coupling between the members and the leader.

The following demonstrations show five characters, where one of them represents the leader (denoted in dark gray). This leader is coupled unidirectionally to all members of the group. If the leader is not present, the group itself ($\alpha = 0$) shows exponential convergence for the coupling strength $k > 1/4$.

Example 6: Figure 6.7, present the case, where $k = 0.01$ and $\alpha = 0$ and the contracting condition is not fulfilled. The system is non-contracting and no coordinated behavior is achieved in the simulation. The group members do not adjust their step cycles to the others nor to the leader.

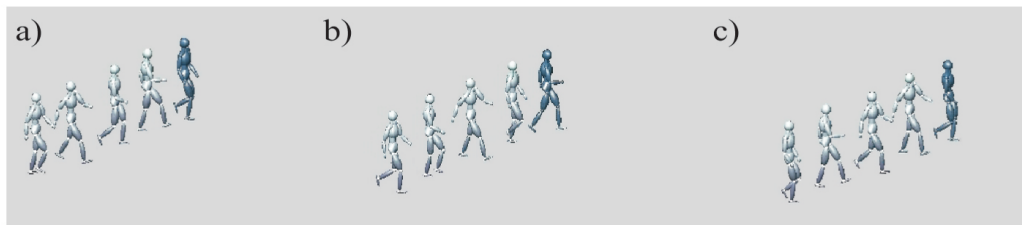


Figure 6.7: Leader-group interaction: The contraction condition is not fulfilled ($k = 0.01$, $\alpha = 0$), showing no coordinated behavior. The characters start with uncoordinated, individual step-phases (a) and do not synchronize (b, c). [Movie_6]

Example 7: If a leader with sufficiently strong coupling is introduced to the other group members ($\alpha = 1$), the theoretical contraction bound is fulfilled (Figure 6.8). During the time interval, a fast convergence to a coordinated behavior can be observed. Even for small values of the coupling strength k within the group, the whole system including the leader oscillator (character) is contracting.

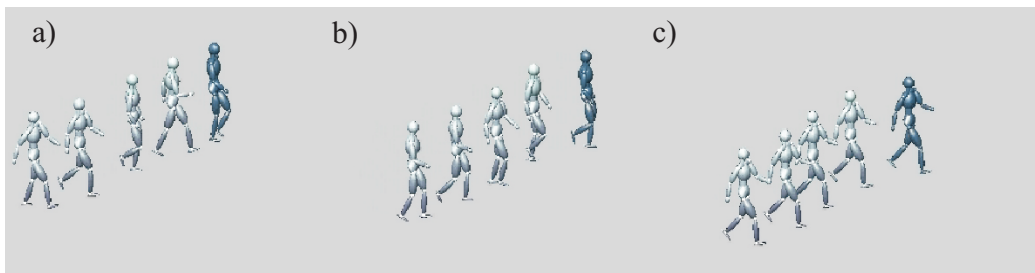


Figure 6.8: Leader-group interaction with a fulfilled contraction condition. A synchronization behavior can be observed. [Movie_7]

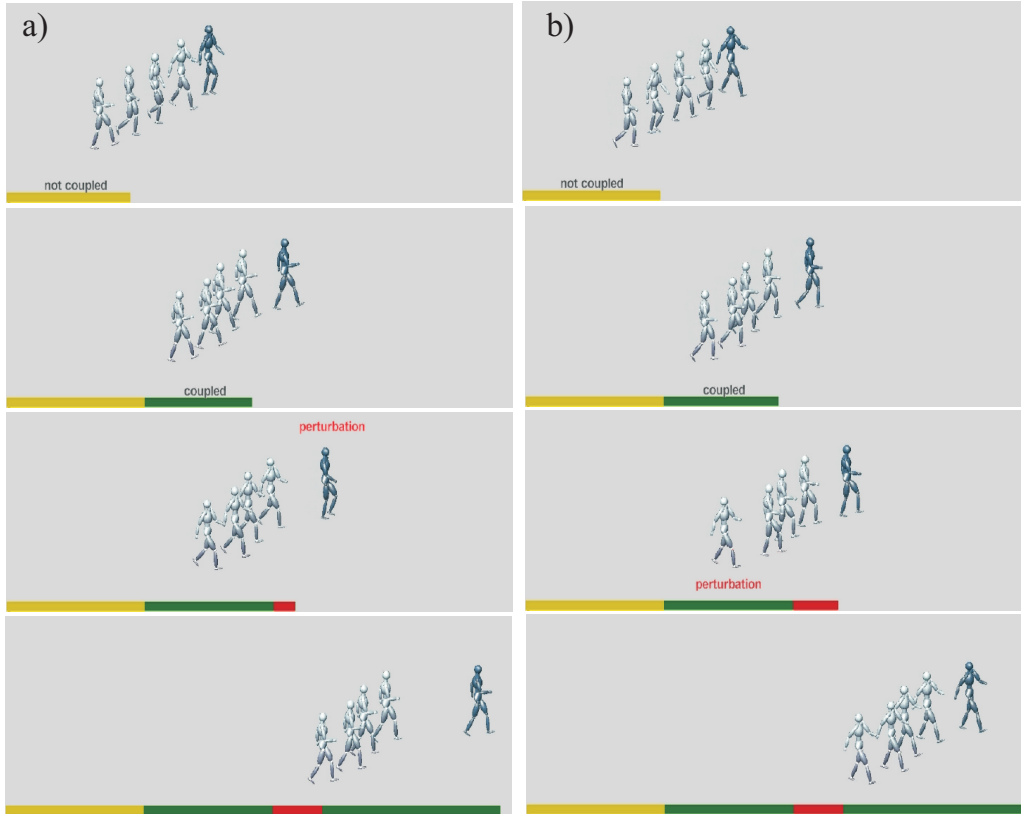
Example 8:

Figure 6.9: Leader-group interaction: a) At the beginning, the characters start with the initial condition (yellow bar). After introducing couplings the group becomes coordinated (green bar). The leader experiences a perturbation by changing its frequency (red bar) and the follower adapts to the leader's phase again. b) Instead of the leader, a member of the group gets perturbed in an identical way and relaxes back to reestablish synchrony with the other members of the group. [Movie_8] and [Movie_9]

The last Figure 6.9 corresponds to the case, where the group members are coupled within the group with $k = 0.2$ and to the leader with the gain of $\alpha = 0.25$, while fulfilling the theoretical bound for contraction. The characters converge very quickly to a coordinated behavior. Additionally, perturbations to one of the followers and to the leader were included in order to see contraction behavior more properly.

In the first interval of Figure 6.9 a), marked with a yellow bar, the characters are initially uncoupled and start with random initial phases. After activating the

coupling (green bar), the group converges to the same state. In the next interval, denoted by the red bar, the leader experiences a phase perturbation by changing the frequency of the oscillator. The group quickly adopts the behavior of the leader and ends with the same common frequency, which can be seen in the last interval. The panel b) of this figure represents a similar example and illustrates a perturbation of the same size, which is not applied to the state of the leader but to the one of a group member by again changing the frequency. It can be observed that its behavior quickly relaxes back to reestablish synchrony with the other members of the group.

6.1.4 Theoretical vs. Empirical Convergence Rates

As a more systematic validation of the theoretical bounds, we also computed empirical convergence rates

$$\lambda^{\text{exper}} = 1/\tau^{\text{exper}}$$

for groups of characters of different size N . These rates were obtained by assuming approximately exponential convergence of the sizes of virtual displacements:

$$||\delta x|| \sim e^{-t/\tau^{\text{exper}}}$$

The norm of the virtual displacements was approximated by the angular dispersion [Kur84]:

$$\hat{R} = \left(1 - \frac{1}{N} \left| \sum_j e^{i\phi_j} \right| \right)^{\frac{1}{2}}$$

of the phases ϕ_j of the oscillators, averaged over 100 simulations with random initial conditions.

Figure 6.10 a) shows the logarithm of this dispersion measure as a function of time (in gait cycles). It shows an initial constant segment (offset time), and after that a nearly linear decay with time, from which the time constant τ^{exper} can be estimated by linear regression. Figure 6.10 b) shows the offset times as a function of the coupling strength to different types of coupling.

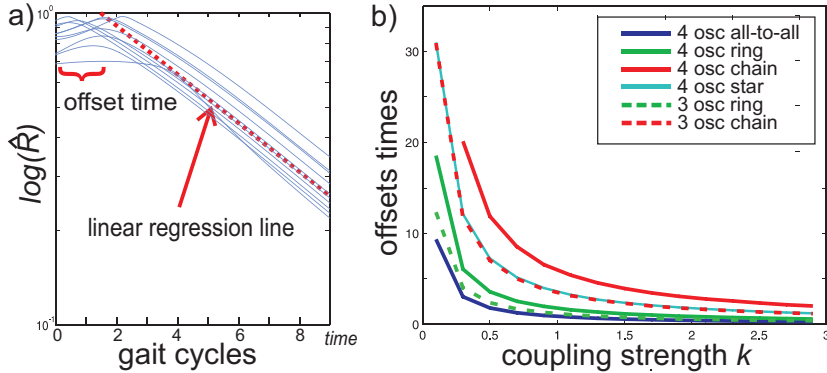


Figure 6.10: a) Dispersion of the phase of the oscillators, averaged over 100 simulations with random initial conditions, as function of time (gait cycles). After an offset time, during which the dispersion remains relatively constant, it decays exponentially. Convergence rates were estimated by fitting linear functions to this decay. b) Offset times (in gait cycles) as function of the coupling strength. (End of offset time interval was defined by the point where the regression line crosses the level $\hat{R} = 1$.)

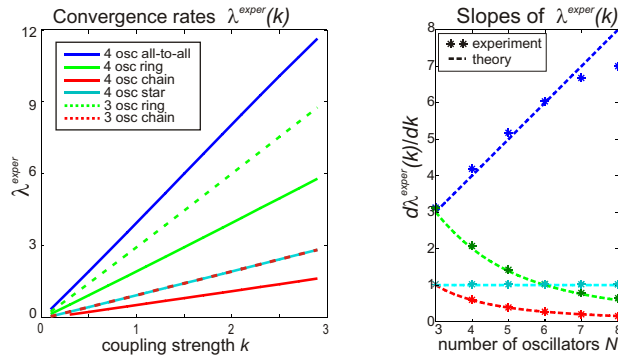


Figure 6.11: a) The relationship between convergence rate and coupling strength k for different types of coupling graphs. b) Slopes of this relationship as function of the number N of Hopf oscillators, comparing simulation results (indicated by asterisks) and derived from the theoretical bounds (Section 2.4.1).

Figure 6.11 a) illustrates the dependency between coupling strengths k and the convergence rate λ^{exper} as estimated from simulations in the regime of the exponential convergence. As derived from the theoretical bound, the convergence rate varies linearly with the coupling strength. In the case of three oscillators, the ring coupling is equivalent to the all-to-all coupling. Figure 6.11 b) shows the slope $d\lambda^{\text{exper}}(k)/dk$ of this linear relationship as function of the number of oscillators N

in the network.

We find a close similarity between the theoretically predicted relationship (dashed curves) and the results from the simulation (indicated by the stars). Indeed, it is evident that for all-to-all coupling, the convergence rate increases with the number of oscillators; while for chain or ring coupling, the convergence speed decreases with the number of oscillators (for a fixed coupling strength). These results show in particular that the proposed theoretical framework is not only suitable for proving asymptotic stability, but also for guaranteeing the convergence speed of the system dynamics.

6.1.5 Distance Frequency Coupling

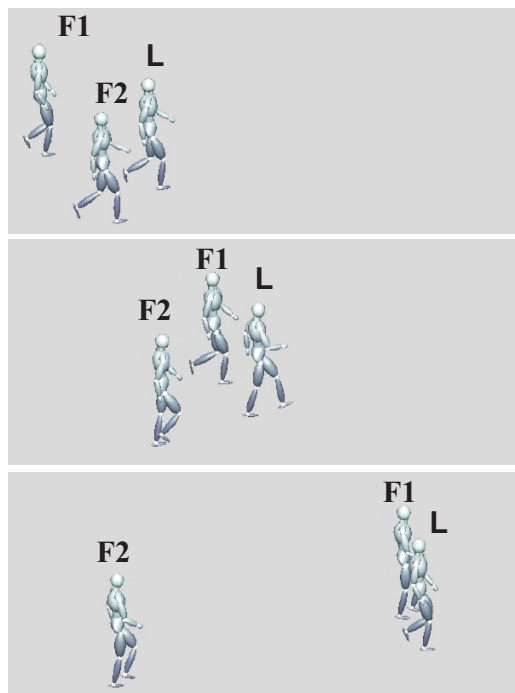


Figure 6.12: Distance Frequency Coupling: Follower $F1$ was coupled with positive and follower $F2$ with a negative coupling force to the leader L , whose frequency is dependent on the distance. $F1$ converges to the leader by increasing the speed, where the behavior of $F2$ diverges to minus infinity, which shows a back walk.[[Movie_10](#)]

The last part of this section shows the results of the contracting conditions for the distance-frequency coupling in a following behavior (see Section 2.5). We de-

rived the equation of the form $\omega(t) = \omega_0 + m(z_1(t) - z_2(t))$ where m defines the coupling strength. We assume that ω_0 is the frequency of the follower, which corresponds to the propagation speed of the Leader (ω_0 can be estimated by observing the Leader). This means that, depending on the coupling force, we obtain an automatically following behavior, which is presented in Figure 6.12. Three characters are animated, where two of them characterize the followers ($F1$ and $F2$), whose frequency is dependent on the distance to the third one, the leader (L). To show the different behaviors dependent on the coupling strength, we coupled one follower with a positive and the other with a negative coupling force. If $m > 0$, the position of the follower converges to the leader by increasing the eigenfrequency. In the contrary case, ($m < 0$), the behavior diverges exponentially to minus infinity, which shows a backward-walking comportment.

CHAPTER 7

Testing Environment

This chapter briefly presents the main software tools that have been used for our animation engine. The proposed methods and formulas were implemented in Matlab (Matrix Laboratory) R2007b, an appropriate choice for the solution of numerical computation. In addition to this, it serves as an excellent environment to develop and test algorithms, which have been already discussed in detail throughout the previous chapters.

7.1 Visualization

The output of the motion synthesis was provided in terms of a standard marker set and standard skeleton, which have been animated in three different ways: In Matlab, 3D Studio Max (Autodesk Inc.), and in the Graphics Engine Horde3D.

MATLAB: Besides serving as the environment for implementation, Matlab represents the testing platform for the visualization of the generated motion trajectories. For this purpose, simple puppet figures were animated using the information of the marker position to attach geometrical shapes –ellipsoids and cylinders– to the marker positions, whereby the apex of an ellipsoid represents the joint centers (Figure 7.1). A more detailed description of the avatar modeling in Matlab can be found in [Oml10].

3D Studio Max (Autodesk, Inc.) is a modeling and animation software for obtaining a more realistic-looking scenario and has become a standard tool for video game developers and other visualization studios. Since the marker set of the com-

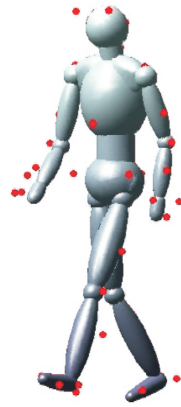


Figure 7.1: Modeling avatar: Geometrical shapes were attached to the computed marker positions.

puted motion trajectories are compatible with 3D Studio Max, they were imported in CSM file format -an optical tracking format for importing marker data- in the animation software through its plug-in architecture. The character models we used were exported to 3D Studio Max from a commercial set of human avatars ('Complete Characters' from Rocketbox Studios GmbH), especially created for real-time environments, e.g. video games, and providing a complete bone skeleton and skinning. Our trajectory set had to be retargeted on the chosen characters by using a 3D Studio Max model-matching technique to obtain joint angle trajectories. Therefore, character rigs or 'Bipeds', illustrated in Figure 7.2, had to be pre-made for animation purposes, which represent the standard skeleton and are needed to adjust our motion trajectories to the desired character. In this way, more sophisticated characters, whose movements were then described by the synthesized motion data, can be merged together to animate the desired scenario (Figure 7.3).

Moreover, for establishing an interchange file format for interactive 3D applications with the ability to transfer the character information into our Graphics Engine Horde3d, we converted them into the COLLADA (COLLABorative Design Activity) format (*.DAE). COLLADA defines an open standard XML schema for exchanging digital assets among various graphics software applications that might otherwise store their assets in incompatible file formats. Through this format we were able to transport the body information into the Graphics Engine.

Graphic Engine: Horde3D rendering engine is a cross-platform compatible by using OpenGL as rendering API. It is a powerful graphics engine comprising

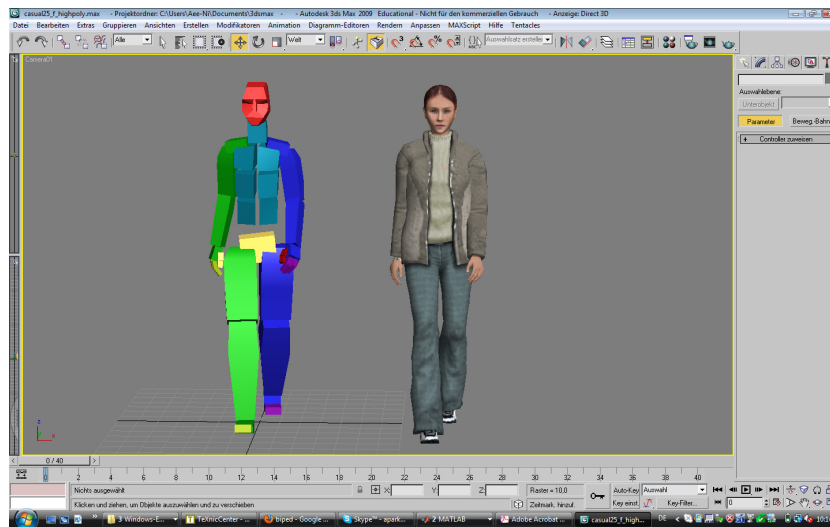


Figure 7.2: Motion trajectories have to be mapped onto a Biped 3D-model (left), which defines the animation of a character (right).

modern shader-based architecture (automatic shader permutation generation using GLSL shaders, OpenGL Shading Language, fully supported by NVidia GPU). It is possible to create an online polygonal resolution adaptation by software skinning and GPU hardware skinning in vertex shader for rendering hundreds of animated characters. It supports: High Dynamic Range (HDR) textures and lighting; almost all modern rendering techniques, including parallax mapping; real-time shadows using Parallel Split Shadow Maps (PSSM) which takes into account hardware acceleration. Data-driven rendering pipelines for straight switching between different rendering techniques includes optimization of geometry for GPU post-transform vertex cache. Horde3D is programmed in object-oriented C++ code and is accessible through a procedural C-style interface similar to the Microsoft Win32 API. This interface provides routines for loading data from files, streams or any type of archives, mixing of binary and XML formats for better trade-offs between performance and productivity. A Collada Converter links Horde 3D with many common Digital Content Creation (DCC) tools. The Horde3D rendering engine provides access to joint data for dynamic animations and ragdoll physics and access to vertex data for collision detection and interoperability with physics engines. The unified low-level animation system is working directly on scene graph.

Collada Converter of Horde3D reshapes the data from Collada-format files into assets accessible by Horde3D, creating separate files:

1. A scene graph resource (XML document) defining branch of the scene graph

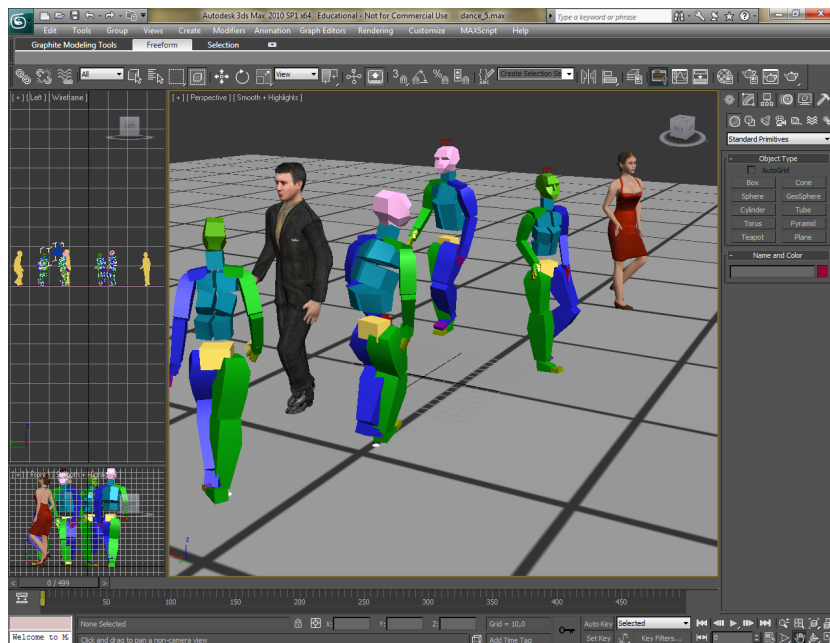


Figure 7.3: This screenshot shows an animation of a dance scenario in 3D-Studio Max. Characters contain Biped models (bone skeleton), which comprise the motion information

and its complete hierarchy in a form providing access to the components of a model like meshes and joints using the scene graph API.

2. A geometry file with polygonal data used by models and meshes containing triangle indices and several streams for vertex attributes like texture coordinates etc.
3. An animation file for joints containing the local transformation for each frame in the form of a translation vector, a scale vector and a rotation quaternion.

Only the first two components were imported while the third was modified online by the developed online animation software. We created several standalone programs in C++ for animating offline produced scenarios using Horde3D. Moreover, we established an interface between Horde 3D and the Matlab psychophysics toolbox for online control of the engine.

In order to provide an online reactive control together with a timing control for real-time rendering engine embedding (like Horde 3D) in Matlab programs, we established in collaboration with the Max Planck Institute for Biological Cybernetics, Tuebingen, an interface between Horde3D and the MATLAB Psychophysics Toolbox v.3 (psychtoolbox.org). It provides the user with a high-level interpreted language (Matlab) with a well-defined interface to the graphics hardware that ac-

cesses the display frame buffer and color lookup table, allows synchronization with the vertical blanking, supporting millisecond timing, and facilitates the collection of observer responses.

The interface was programmed as C++ MEX file, a wrapper around the C/C++ interface of the Horde3D graphics engine, which allows addressing Horde 3D commands directly from Matlab with a well-controlled timing. The interface uses the Matlab Psychtoolbox-3 that coordinates its access to an OpenGL rendering engine with Horde 3D. This resulting framework combines the advantages of the Matlab Psychophysics Toolbox, which makes it possible for experimentalists to program easily experiments with highly controlled timing of stimuli and response acquisition, and the advantages of Horde 3D, which provides a variety of tools for the real-time simulation of crowds, rendering with variable resolution, etc. The developed interface is compiled with the core Horde3D engine and gives access to all Horde3D classes and functions accessing vertex and joint data. This is specifically important for the online generation of trigger events and enforcing kinematic constraints derived from the interaction between individual characters and the scene. The interface plug-in also dynamically uploads Horde3D assets and resources.

CHAPTER 8

Conclusion and Outlook

Animating virtual humans and, in particular the real-time simulation of interactive crowd behavior is a challenging task in computer animation for film and game applications. In this context, one of the core problems is the synthesis of believable human motion, a complex procedure in which many joints and its redundant degrees-of-freedom are involved. This high dimensional volume of control information for coordinating the motion of an articulated figure makes character animation tedious.

Furthermore, to develop a real-time capable system that interacts with the dynamic environment or other articulated figures, immediate interaction skills have to be considered, such as taking into account navigating through moving and stationary obstacles. In addition to this, different convincing emotional changes might be included for a more realistic scenario. The outcome of this mentioned procedure appears as a complex architectural system containing many nonlinear subsystems, in which the treatment can be considered sophisticated and infeasible. Given the intricacy of this background, the goal of our framework was the development of learning-based methods for the synthesis of highly realistic human movements in real-time.

The proposed method is biologically inspired and exploits the concept of synergies, which is derived from motor control. Synergies specify lower-dimensional control units that encompass only a subset of the available degrees of freedom. It has been suggested that complex motor behavior, characterized by a large number of redundant degrees of freedom, is controlled by a superposition of such syn-

ergies. In addition to this, work in motor control has tried to identify synergies from trajectory and EMG data, applying unsupervised learning. We used a similar approach to solve the degrees of freedom problem in full-body animation and developed an algorithm for real-time animation, which can be summarized in the following steps:

1. Learned movement primitives, or ‘synergies’, were extracted from full-body motion capture data and comprise periodic gait movements with simple non-periodic arm movements. Therefore, we applied an algorithm for blind source separation, based on ICA with time-delays, and modeled the data by mixing delayed sources (anechoic mixtures). The described model for the compact approximation of trajectories based on synergies is not suitable for real-time animation, since the trajectories have to be synthesized by superposition and delaying of source signals whose whole time-course must be known.
2. For this reason, we obtained a real-time capable animation system in which we defined nonlinear dynamical systems that generate the associated trajectories online. This is done by constructing a nonlinear mapping between the attractor solutions of the dynamical systems (i.e. VdP oscillators) and the source signals using Support Vector Regression (SVR). In this way, complex kinematics are described by simple non-linear dynamical systems.
3. Then, the character is driven by the non-linear dynamical systems that correspond to the extracted movement primitives, which we coupled to obtain a stabilized coordinated behavior. Additionally, we included morphing algorithms, so that various numbers of motion styles with different emotions can be realized by blending among corresponding morph parameters.
4. The introduction of appropriate dynamic couplings among the dynamical systems controlling each individual avatar, allows the generation of a coordinated or synchronized group behavior. The developed framework can now be easily combined with key elements of real-time animation systems, such as style morphing and obstacle avoidance. Therefore, navigation dynamics had to be developed and can be used for interactive behavior with the dynamic environment. A variety of application scenarios, such as following behaviors –involving speed and distance control algorithms– could be demonstrated.
5. It is usually a difficult task to analyze the stability of such coupled time-varying systems. Thus, we presented a new approach to design stability

properties of animation systems that generate collective behavior of characters through self-organization from interacting dynamical models. In contrast with many existing approaches, our system uses simple non-linear dynamical systems, as opposed to more detailed biomechanical or physical models. Such systems are in principle accessible for an application of tools from stability theory. We have also introduced contraction analysis as a new framework for the study and to design the stability properties of online animation systems. The proposed theoretical approach has the advantage of deriving global stability results for nonlinear systems from local stability properties that can be easily verified. Contraction theory also offers the possibility to treat the stability of complex systems since it permits to transfer stability results from components to composite systems. Many other approaches for stability analysis do not have this property, which makes the analysis of complex systems often intractable. Moreover, we also demonstrated that this theory is suitable to compute bounds for the convergence rate of such systems, where sufficiently fast convergence is important for many applications.

Although our approach is a fundamental step in the right direction, it has a lot of potential to be extended and to be improved in different kinds of aspects. Few of them can be summarized in the following points:

- Future work will extend this approach for more complex movements, and will focus on exploiting more the concept of synergy, trying to learn sparse components that encompass only limited sets of degrees of freedom. This will potentially result in a more flexible control of motion styles.
- In fact, including feedback control will offer the possibility to make the articulated figures or rather individual 'synergies' reactive and dependent on external constraints, potentially providing a basis for a much more fine-grained adaptation of the generated behavior to external constraints, which would be also interesting for other research field; possible applications in robotics will be considered.
- Another interesting task would be a systematic investigation with other methods by comparing ground truth data for movements in interactive scenarios, including psychophysical experiments. Psychophysical experiments can be used to classify the extend of the realism of the animated scenario and the applications can be improved to obtain a more realistic acceptance rate for the observer.
- Self-organized crowd animation can be relevant for advertisement proposals and for simulation of safety procedures, e.g. the evacuation process in

airplanes.

- Finally, since in this work we tried to sketch some first steps towards a development of a systematic approach for the design of the dynamics of such online animation systems, future work needs to evaluate whether this work can be extended to more complex systems that combine periodic and non-periodic primitives and other dynamical components, such as navigation, or the arbitration of different behaviors.

Appendix

.1 Notation

T	Time pedriod
ϕ	Phase of an oscillation
π	ratio of circle's circumference to its diameter
ω	Frequency
a	amplitude
ζ	Speed control (for amplitude damping)
r	Radius
\mathbf{J}	Jacobian
$(\cdot)_s$	Symertical part
λ	Eigenvalue
λ_L^+	Positive Eigenvalue of the Laplacian
δx	virtual displacement
sup	Supremum
θ	Coordinate transform
\mathbf{F}	Generalized Jacobian
M	Contracting metric
\mathcal{M}	Flow invariant manifold
I	Identity matrix
A^T	Transpose of the matrix A
A^{-1}	Inverse of the matrix A
k, α	Coupling gains
\mathcal{N}_i	Set of neighbors of i
L	Laplacian matrix
L_G	Laplacian of coupling graph
Δ	Gradient

$g(\cdot)$	Nonlinear function
ξ	Nonlinear coupling
w	Mixing weight
s	Sources signal
m	Mean
τ	Time delay
$E\{\cdot\}$	Expected value
$\arg(z)$	Complex argument of the complex number z
$\text{trace}(A)$	Trace of the matrix A
\tanh	Hyperbolic tangent
C	Covariance matrix
$\ x\ _F$	denotes the Frobenius norm of x .
\log	Natürlicher logarithm (e als Basis)
\circ	Hadamard or Schur product. Formally, for two matrices of the same dimensions: $A, B \in \mathbb{R}^{m \times n}$ the Hadamard product is a matrix of the same dimensions $A \circ B \in \mathbb{R}^{m \times n}$ with elements given by $(A \circ B)_{i,j} = A_{i,j} \cdot B_{i,j}$
\mathcal{F}	Fourier transform
\mathcal{F}^{-1}	Inverse Fourier transform
\mathbf{H}	orthogonal matrix
v	Dynamic noise
φ	Heading direction
η	Morphing parameter
d	Distance
\mathbf{v}	Momentary velocity
\hat{R}	Angular dispersion
$\rho, \lambda^{\text{exper}}$	convergence rate

.2 Tables

Table 1: Marker names and definitions of the Vicon Plug-In Gait model

Marker Name	Definition	Marker Name	Definition
Upper Body			
LFHD	Left front head	RFHD	Right front head
LBHD	Left back head	RBHD	Right back head
C7	Seventh cervical vertebrae	T10	Tenth thoracic vertebrae
CLAV	Clavicle	STRN	Sternum
LSHO	Left shoulder	RBAC	Right back
LUPA	Left upper arm	RSHO	Right shoulder
LELB	Left elbow	RUPA	Right upper arm
LFRM	Left forearm	RELB	Right elbow
LWRA	Left wrist A	RFRM	Right forearm
LWRB	Left wrist B	RWRA	Right wrist A
LFIN	Left finger	RWRB	Right wrist B
		RFIN	Right finger
Lower Body			
LASI	Left front waist	RASI	Right front Waist
LPSI	Left back waist	RPSI	Right back Waist
LTHI	Left thigh	RTHI	Right thigh
LKNE	Left knee	RKNE	Right knee
LTIB	Left shin	RTIB	Right shin
LANK	Left ankle	RANK	Right ankle
LHEE	Left heel	RHEE	Right heel
LMT5	Left 5th metatarsal	RMT5	Right 5th metatarsal
LTOE	Left toe	RTOE	Right toe

Table 2: Skeleton segments, and attached coordinate frames $[x, y, z]$

Segment	Marker	Joint (center of rotation)	x	y	z
Pelvis	L/RASI				
	L/RPSI	$\frac{LASI+RASI+LPSI+RPSI}{4}$	$\frac{LASI-RASI}{\ LASI-RASI\ }$	$\frac{LASI+RASI-LPSI-RPSI}{\ LASI+RASI-LPSI-RPSI\ }$	$x \times y$
	L/RFEO				
Thorax_null					
C7					
T10					
CLAV					
Thorax	STRN			$\frac{LSHO-RSHO}{\ LSHO-RSHO\ }$	$\frac{C7+CLAV-STRN-T10}{\ C7+CLAV-STRN-T10\ }$
	L/RSJC	Thorax_null= $\frac{STRN+T10}{2}$	$y \times z$		
	L/RSHO				
	RBAK				
	NECK				
Head	L/RFHD	NECK= $\frac{C7+CLAV}{2}$	$\frac{LFHD+LFHD-LBHD-RFHD}{\ LFHD+LFHD-LBHD-RFHD\ }$	$\frac{LFHD-RFHD}{\ LFHD-RFHD\ }$	$x \times y$
	L/RBHD				
L/R	L/RUPA				
upper Arm (Humerus)	L/RELB	L/RSJC	$y \times z$	$\pm \frac{L/RELB-L/REJC}{\ L/RELB-L/REJC\ }$	$\frac{L/RSJC-L/REJC}{\ L/RSJC+L/REJC\ }$
	L/REJC				
L/R	L/RFRA				
lower Arm	L/RWRA	L/REJC	$y \times z$	$\pm \frac{L/REJC-L/RELB}{\ L/REJC-L/RELB\ }$	$\frac{L/REJC-L/RWJC}{\ L/REJC+L/RWJC\ }$
	L/RWRB				
(Radius)	L/RWJC				
L/R Hand	L/RFIN	L/RWJC	$y \times z$	$\pm \frac{L/RWRA-L/RWJC}{\ L/RWRA-L/RWJC\ }$	$\frac{L/RWJC-L/RFIN}{\ L/RWJC-L/RFIN\ }$
L/R	L/RTHI				
upper Leg (Femur)	L/RKNE	L/RFEP	$y \times z$	$\pm \frac{L/RFEO-L/RKNE}{\ L/RFEO-L/RKNE\ }$	$\frac{L/RFEP-L/RFEO}{\ L/RFEP-L/RFEO\ }$
	L/RFEO				
L/R	L/RTIB				
lower Leg (Tibia)	L/RANK	L/RFEO	$y \times z$	$\pm \frac{L/RANK-L/RTIO}{\ L/RANK-L/RTIO\ }$	$\frac{L/RFEO-L/RTIO}{\ L/RFEO-L/RTIO\ }$
	L/RTIO				
L/R	L/RMT5				
Foot	L/RTOE	L/RTIO	$\frac{L/RTOE-L/RHEE}{\ L/RTOE-L/RHEE\ }$	$\pm \frac{L/RANK-L/RTIO}{\ L/RANK-L/RTIO\ }$	$x \times y$
	L/RHEE				

.3 Authored Publications

Park A., Mukovskiy A., Slotine J. J. E., Giese M. A.: Design of dynamical stability properties in character animation. In: The 6th Workshop on Virtual Reality Interaction and Physical Simulation. ,VRI-PHYS 09, Nov 5-6, Karlsruhe, Proc., pp.85-94

Giese M. A., Mukovskiy A., Park A., Omlor L., Slotine J. J. E.: Real-Time Synthesis of Body Movements Based on Learned Primitives. In Cremers D, Rosenhahn B, Yuille A L (eds): 'Statistical and Geometrical Approaches to Visual Motion Analysis', Lecture Notes in Computer Science, 5604, pp. 107-127. Springer Verlag.

Mukovskiy A., Park A., Omlor L., Slotine J. J. E., Giese M. A.: Self-organization of character behavior by mixing of learned movement primitives. Proceedings of the 13th Fall Workshop on Vision, Modeling, and Visualization (VMV), October 8-10, Konstanz, Germany.

Park A., Mukovskiy A., Omlor L., Giese M. A.: Self organized character animation based on learned synergies from full-body motion capture data. International Conference on Cognitive Systems (CogSys), Springer-Verlag, Berlin.

Park A., Mukovskiy A., Omlor L., Giese M. A.: Synthesis of character behavior by dynamic interaction of synergies learned from motion capture data. Skala V. (ed): Proceedings of the 16th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG), Plzen, Czech Republic, pp. 9-16.

Park A., Omlor L., Giese M. A.: Synergy-based method for the self-organization of full-body movements with high degree of realism. Bülthoff H. H., Chatziastos A., Mallot H. A., Ulrich R. (eds): Proceedings of the 10th Tuebinger Perception Conference (TWK 2007), Knirsch, Germany, pp. 152.

Bibliography

- [AF02] O. Arikan and D. A. Forsyth. Interactive motion generation from examples. *ACM Trans. on Graphics, SIGGRAPH '02*, 21(3):483–490, 2002. [1.3.1.1](#)
- [AFO03] O. Arikan, D. A. Forsyth, and J. F. O'Brien. Motion synthesis from annotations. *ACM Trans. on Graphics, SIGGRAPH '03*, 22(3):402–408, 2003. [1.3.1.1](#), [1.3.1.4](#)
- [AHS03] I. Albrecht, J. Haber, and H. P. Seidel. Construction and animation of anatomically based human hand models. *Proc. ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA 2003)*, pages 98–109, 2003. [1.3.3](#)
- [AMS97] C. G. Atkeson, A. W. Moore, and S. Schaal. Locally weighted learning for control. *AI Review*, 11:75–113, 1997. [6](#)
- [ASDB08] M. A. Azahar, M. S. Sunar, D. Daman, and A. Bade. *Survey on Real-Time Crowds Simulation*. Springer-Verlag, Berlin/Heidelberg, 2008. [1.3.1.2](#)
- [AVK87] A. A. Andronov, A. A. Vitt, and S. E. Khaikin. *Theory of Oscillators*. Dover Publication Inc., New York, 1987. [2.2.4](#), [2.4.1](#)
- [BB07] D. Bouchard and N. Badler. *Semantic Segmentation of Motion Capture Using Laban Movement Analysis*. Springer-Verlag, Berlin/Heidelberg, 2007. [1.3.1.1](#)

- [BCRP97] B. Bodenheimer, R. Charles, S. Rosenthal, and J. Pella. The process of motion capture. *In: Computer Animation and Simulation'97*, 52(5):3–18, 1997. [1.3.1.1](#)
- [Ber67] N.A. Bernstein. *The Coordination and Regulation of Movements*. Pergamon Press, New York, 1967. [1.3.2.1](#)
- [BH97] D. C. Brogan and J. K. Hodgins. Group behaviors for systems with significant dynamics. *Autonomous Robots*, 4(1):137–153, 1997. [1.3.3](#)
- [Bis07] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 2007. [3.2.1](#)
- [BRI06] J. Buchli, L. Righetti, and A. J. Ijspeert. Engineering entrainment and adaptation in limit cycle systems from biological inspiration to applications in robotics. *Biological Cybernetics*, 95(6):645–664, 2006. [1.3.3](#), [4.3.2](#), [6](#)
- [BS95] A. J. Bell and T. J. Sejnowski. An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7:1129–1159, 1995. [3.1](#)
- [BW95] A. Bruderlin and Williams. Motion signal processing. *In: Computer Graphics*, 29(Annual Conference Series):97–104, 1995. [1.3.1.1](#)
- [CBGI05] A. Crespi, A. Badertscher, A. Guignard, and A. Ijspeert. Amphibot i: An amphibious snake-like robot. *Robotics and Autonomous Systems*, 50(4):163–175, 2005. [1.3.2.2](#)
- [CCPL05] S. Choi, A. Cichoski, H. H. Park, and S. Y. Lee. Blind signal separation and independent component analysis: A review. *Neural Information Processing*, 5:1–57, 2005. [3.2.2](#)
- [CDF⁺01] S. Camazine, J. L. Deneubourg, N. R. Franks, J. Sneyd, G. Theraulaz, and E. Bonabeau. *Self-Organization in Biological Systems*. Princeton University Press, New Jersey, 2001. [1.3.3](#)
- [CH05] J. Chai and J.K. Hodgins. Performance animation from low-dimensional control signals. *ACM Trans. on Graphics, SIGGRAPH '05*, 24(3):686–696, 2005. [1.3.1.3](#), [1.3.1.4](#)
- [CH07] J. Chai and J.K. Hodgins. Constraint-based motion optimization using a statistical dynamic model. *SIGGRAPH '07: ACM SIGGRAPH 2007*, page 8, 2007. [1.3.1.2](#), [1.3.1.4](#)

- [CL01] C. C. Chang and C. L. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>. [4.3.3](#)
- [Cou09] I. D. Couzin. Collective cognition in animal groups. *Trends in Cognitive Sciences*, 13(1):1–44, 2009. [1.3.3](#), [5.2.2](#), [6](#)
- [CR94] J. J. Collins and S. A. Richmon. Hard-wired central pattern generators for quadrupedal locomotion. *Biological Cybernetics*, 71(5):375–385, 1994. [1.3.2.2](#)
- [CS07] F. Cucker and S. Smale. Emergent behavior in flocks. *IEEE Trans. Automat. Control*, 52(5):852–862, 2007. [1.3.3](#), [6](#)
- [CTF01] T. W. S. Chow, H. Tan, and Y. Fang. Nonlinear system representation. *Wiley Encyclopedia of Electrical and Electronics Engineering*, 2001. [2](#)
- [dB05] A. d’Avella and E. Bizzi. Shared and specific muscle synergies in neural motor behaviours. *Proc Natl Acad Sci*, 102(8):3076–3081, 2005. ([document](#)), [1.5](#), [1.3.2.1](#)
- [dSAP08] M. da Silva, Y. Abe, and J. Popović. Interactive simulation of stylized human locomotion. *ACM Trans. Graph.*, 27(3):1–10, 2008. [1.3.1.3](#)
- [dSB03] A. d’Avella, P. Saltiel, and E. Bizzi. Combinations of muscle synergies in the construction of a natural motor behavior. *Nature Neuroscience*, 6(3):300–308, 2003. [1.3.2.1](#)
- [FH05] T. Flash and B. Hochner. Motor primitives in vertebrates and invertebrates. *Curr. Opin. Neurobiol.*, 15(6):660–666, 2005. [1.3.2.1](#), [3](#), [6](#)
- [FMJ02] A. Fod, M. J. Mataric, and O. C. Jenkins. Motor primitives in vertebrates and invertebrates. *Autonomous Robots*, 12(1):39–54, 2002. [4.3.2](#)
- [GKM⁺01] S. Goldstein, M. Karavelas, D. Metaxas, L. Guibas, E. Aaron, and A. Goswami. Scalable nonlinear dynamical systems for agent steering and crowd simulation. *Computers & Graphics* 25, 25(6):983–998, 2001. [1.3.1.2](#)
- [Gle98] M. Gleicher. Retargetting motion to new characters. *Proc. ACM SIGGRAPH ’98*, pages 33–42, 1998. [1.3.1.1](#)

- [GMHP04] K. Grochow, S. L. Martin, A. Hertzmann, and Z. Popović. Style-based inverse kinematics. *ACM Trans. Graph.*, 23(3):522–531, 2004. [1.3.1.4](#)
- [GRIL08] A. Gams, L. Righetti, A. J. Ijspeert, and J. Lenarcic. A dynamical system for online learning of periodic movements of unknown waveform and frequency. *In Proc. of the sec. IEEE RAS / EMBS International Conference on Biomedical Robotics and Biomechatronics*, 2008. [1.3.2.2](#), [6](#)
- [GSKJ03] M. Gleicher, H. J. Shin, L. Kovar, and A. Jepsen. Snap-together motion: Assembling run-time animation. *ACM Trans. on Graphics, SIGGRAPH '03*, 22(3):702–702, 2003. [1.3.1.1](#)
- [GTH98] R. Grzeszczuk, D. Terzopoulos, and G. Hinton. Neuroanimator: Fast neural network emulation and control of physics based models. *Int. Conf. on Comp. Graph. and Interactive Techniques, Proc. ACM SIGGRAPH'98*, 61(5):9–20, 1998. [1.3.1.2](#), [1.3.3](#), [6.1](#)
- [HbF09] C. Hosted by Freistetter. Tornado unter wasser. 2009. ([document](#)), [1.6](#)
- [HbT08] K. Hosted by Taylor. Gollum, lord of the rings trilogy. 2008. ([document](#)), [1.1](#)
- [HO01] A. Hyvaerinen and E. Oja. Independent component analysis: Algorithms and applications. *Neural Networks*, 13(4-5):411–430, 2001. [3.1](#), [3.2.2](#)
- [HPP05] E. Hsu, K. Pulli, and J. Popović. Style translation for human motion. *ACM Trans. on Graphics, SIGGRAPH'05*, 24(3):1082–1089, 2005. [1.3.1.3](#)
- [Huy67] C. Huygens. *OEvres Complètes*, volume 30. Swets&Zeitlinger B. V., Amsterdam, 1967. ([document](#)), [2.1](#)
- [HWBO95] J. K. Hodgins, W. L. Wooten, D. C. Brogan, and J. F. O'Brien. Animating human athletics. *Proc. ACM SIGGRAPH'95*, pages 71–78, 1995. [1.3.1.2](#), [1.3.3](#), [6.1](#)
- [Hyv99] A. Hyvaerinen. Survey on independent component analysis. *Neural Computing Surveys*, 2:94–128, 1999. [3.2](#), [3.2.1](#), [3.2.2](#)

- [Ijs08] A. J. Ijspeert. Central pattern generators for locomotion control in animals and robots: a review. *Neural Networks*, 21(4):642–653, 2008. [1.3.2.2](#)
- [INS02] A. Ijspeert, J. Nakanishi, and S. Schaal. Learning attractor landscapes for learning motor primitives. *Adv. in Neural Inf. Process Systems*, 15:1547–1554, 2002. [1.3.2.2](#), [4.3.2](#)
- [IPL04] Y. Ivanenko, R. Poppele, and F. Lacquaniti. Five basic muscle activation patterns account for muscle activity during human locomotion. *Journal of Physiology*, 556:267–282, 2004. [1.3.2.1](#), [1.3.2.1](#), [3](#), [3.1.1](#)
- [JM04] O. C. Jenkins and M. J. Matarić. A spatio-temporal extension to isomap nonlinear dimension reduction. *ICML '04: Proc. of the twenty-first international conference on Machine learning*, pages 441–448, 2004. [1.3.1.4](#)
- [KFHT02] H. Kimura, Y. Fukuoka, Y. Hada, and K. Takase. Three-dimensional adaptive dynamic walking of a quadruped - rolling motion feedback to cpgs controlling pitching motion. *2002 IEEE Intl. Conf. on Robotics and Automation (ICRA 2002)*, 3:2228–2233, 2002. [1.3.2.2](#)
- [KG03] L. Kovar and M. Gleicher. Flexible automatic motion blending with registration curves. In: *Proceedings of ACM Symposium on Computer Animation (SCA)*, 2003. [1.3.1.1](#)
- [KS02] A. Kirchner and A. Schadschneider. Simulation of evacuation processes using a bionics-inspired cellular automaton model for pedestrian dynamics. In *Physica A*, 312:260–276, 2002. [1.3.1.2](#)
- [Kur84] Y. Kuramoto. *Chemical Oscillations, Waves, and Turbulence*. Springer-Verlag, Berlin, 1984. [6.1.4](#)
- [Lat08] M. L. Latash. *Synergy*. Oxford University Press, New York, 2008. [1.3.2.1](#)
- [LMIM09] F. D. Libera, T. Minato, H. Ishiguro, and E. Menegatti. Developing central pattern generator based periodic motions using tactile interaction. *9th IEEE-RAS International Conference on Humanoid Robots*, pages 105–111, 2009. [1.3.2.2](#)
- [LS98] W. Lohmiller and J. J. E. Slotine. On contraction analysis for nonlinear systems. *Automatica*, 34(6):683–696, 1998. [2.3](#), [2.3.2](#), [2.3.2](#), [1](#)

- [LS01] D. D. Lee and H. S. Seung. *Algorithms for nonnegative matrix factorization*. MIT Press, Cambridge, MA, 2001. [3.2.3](#)
- [LT02] I. S. Lim and D. Thalmann. Construction of animation models out of captured data. *Multimedia and Expo, 2002. ICME '02*, 1:829–832, 2002. [1.3.1.4](#)
- [LV07] J. A. Lee and M. Verleysen. *Nonlinear dimensionality reduction*. Springer, New York, 2007. [3](#), [3.2](#)
- [Mas70] M. Masahiro. Bukimi no tani the uncanny valley. *Energy*, 7(4):33–35, 1970. [1.1](#)
- [Mat95] M. J. Matarić. Designing and understanding adaptive group behavior. *Adaptive Behavior*, 4(1):51–80, 1995. [1.3.3](#)
- [MBT99] T. Molet, R. Boulic, and D. Thalmann. Human motion capture driven by orientation measurements. *Presence: Teleoper. Virtual Environ.*, 8(2):187–203, 1999. [1.3.1.1](#)
- [Med10] On The Media. The uncanny valley. 2010. ([document](#)), [1.2](#)
- [Men99] A. Menache. *Understanding motion capture for computer animation and video games*. Morgan Kaufmann, California, 1999. [4.1.1](#)
- [MEN⁺06] J. Morimoto, G. Endo, J. Nakanishi, S.-H. Hyon, G. Cheng, D. C. Bentivegna, and C. G. Atkeson. Modulation of simple sinusoidal patterns by a coupled oscillator model for biped walking. *IEEE Intl. Conf. on Robotics and Automation (ICRA 2006)*, page 1579–1584, 2006. [1.3.2.2](#)
- [MLPP09] U. Muico, Y. Lee, J. Popović, and Z. Popović. Contact-aware nonlinear control of dynamic characters. *ACM Trans. on Graphics*, 28(3), 2009. [1.3.1.3](#)
- [Mor81] P. Morasso. Spatial control of arm movements. *Experimental Brain Research*, 42(2), 1981. [4.3.2](#)
- [MT01] S. R. Musse and D. Thalmann. A behavioral model for real time simulation of virtual human crowds. *IEEE Trans. on Visualization and Computer Graphics*, 7(2):152–164, 2001. [1.3.1.2](#), [1.3.3](#)
- [OEH02] P. Oegren, M. Egerstedt, and X. Hu. A control Lyapunov function approach to multi-agent coordination. *IEEE Trans. on Robotics and Automation*, 18(5):847–851, 2002. [1.3.3](#), [6](#)

- [OG06] L. Omlor and M. A. Giese. Blind source separation for over-determined delayed mixtures. *Adv. in Neural Inf. Process Systems*, 19:1049–1056, 2006. [3.3](#), [3.3](#), [4.2.2](#)
- [OG07] L. Omlor and M. A. Giese. Extraction of spatio-temporal primitives of emotional body expressions. *Neurocomputing*, 70:10–12, 2007. [3.3](#)
- [Oml10] Omlor. New methods for anechoic demixing with application to shift invariant feature extraction. *PHD Thesis*, 2010. [3.1](#), [3.1.1](#), [4.1.2.2](#), [7.1](#)
- [OP06] P. O. O’Grady and B. A. Pearlmutter. Convolutional non-negative matrix factorisation with a sparseness constraint. 2006. [3.2.3](#)
- [OPR05] P. O. O’Grady, B. A. Pearlmutter, and S. T. Rickard. Survey of sparse and nonsparse methods in source separation. *International Journal of Imaging Systems and Technology*, 15:18–33, 2005. [3.1](#)
- [PAB⁺06] M. D. Plumbley, S. A. Abdallah, T. Blumensath, M. G. Jafari, A. Nesbit, E. Vincent, and B. Wang. Musical audio analysis using sparse representations. *Compstat 2006 - Proceedings in Computational Statistics*, pages 104–117, 2006. [3.2.3](#)
- [Par02] R. Parent. *Computer Animation: Algorithms and Techniques*. Morgan Kaufmann, California, 2002. [4.1.1](#), [4.1.2.1](#)
- [PLKP07] M. S. Pedersen, J. Larsen, U. Kjems, and L. C. Parra. *A survey of convolutional blind source separation methods*. Springer Press, 2007. [3.1](#)
- [PRK03] A. Pikovsky, M. Rosenblum, and J. Kurths. *Synchronization, A Universal Concept in Nonlinear Sciences*. Cambridge University Press, Cambridge, 2003. [\(document\)](#), [1.3.3](#), [2.1](#)
- [PS07] Q. C. Pham and J. J. E. Slotine. Stable concurrent synchronization in dynamic system networks. *Neural Networks*, 20(3):62–77, 2007. [2.3](#), [2.3.3](#), [2.3.3](#), [2.4](#), [2.4.1](#), [2.4.1.1](#)
- [RCB98] C. Rose, M. Cohen, and B. Bodenheimer. Verbs and adverbs: Multidimensional motion interpolation. *IEEE Computer Graphics and Applications*, 18(5):32–40, 1998. [1.3.1.1](#)

- [Rey99] C. W. Reynolds. Steering behaviors for autonomous characters. *In Proc. of Game Developers Conference 1999*, pages 763–782, 1999. [1.3.3](#)
- [RG06] J. Rose and J. G. Gamble. *Human Walking*. Lippincott Williams and Wilkins, Philadelphia, 2006. [5.1.2](#)
- [RGBC96] C. Rose, B. Guenter, B. Bodenheimer, and M. Cohen. Efficient generation of motion transitions using spacetime constraints. *Int. Conf. on Comp. Graph. and Interactive Techniques, Proc. ACM SIGGRAPH'96*, 30:147–154, 1996. [1.3.1.1](#)
- [RI06] L. Righetti and A. J. Ijspeert. Programmable central pattern generators: an application to biped locomotion control. *Proc. of the 2006 IEEE Int. Conference on Robotics and Automation*, pages 1585 – 1590, 2006. [1.3.2.2](#), [6](#)
- [SB05] A. E. Seward and B. Bodenheimer. Using nonlinear dimensionality reduction in 3d figure animation. *ACM-SE 43: Proc. of the 43rd annual Southeast regional conference*, pages 388–392, 2005. [1.3.1.4](#)
- [SCCH09] T. Shiratori, B. Coley, R. Cham, and J. K. Hodgins. Simulating balance recovery responses to trips based on biomechanical principles. *SCA '09: Proc. of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 37–46, 2009. [1.3.1.3](#)
- [SDE95] G. Schöner, M. Dose, and C. Engels. Dynamics of behavior: Theory and applications for autonomous robot architectures. *Robotics and Autonomous Systems*, 16(2-4):213–245, 1995. [4.3.2](#), [5.2.1](#)
- [SFS98] M. Santello, M. Flanders, and J.F Soechting. Postural hand synergies for tool use. *Journal of Neuroscience*, 18(23):10105–15, 1998. [1.3.2.1](#), [3.1.1](#)
- [SH07] A. Safonova and J. K. Hodgins. Construction and optimal search of interpolated motion graphs. *ACM Trans. on Graphics (SIGGRAPH 2007)*, 26(3), 2007. [1.3.1.1](#)
- [SHP04] A. Safonova, J. K. Hodgins, and N. S. Pollard. Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. *ACM Trans. on Graphics, Proc. SIGGRAPH'04*, 23(3):514–521, 2004. [1.3.1.2](#), [1.3.1.4](#), [4.2](#)

- [SIB03] S. Schaal, A. J. Ijspeert, and A. Billard. Computational approaches to motor learning by imitation. *Philos Trans R Socof London, Biol. Sci.*, 358(1431):537–547, 2003. [1.3.2.1](#), [1.3.2.2](#), [4.3.2](#), [6](#)
- [SKSY08] H. P. H. Shum, T. Komura, M. Shiraishi, and S Yamazaki. Interaction patches for multi-character animation. *Proc. ACM SIGGRAPH Asia 2008*, 27(5):1–b, 2008. [5.2.2](#)
- [SL81] J. F. Soechting and F. Lacquaniti. Invariant characteristics of a pointing movement in man. *Journal of Neuroscience*, 1:710–720, 1981. [4.3.2](#)
- [SL06] H. J. Shin and J. Lee. Motion synthesis and editing in low-dimensional spaces: Research articles. *Comput. Animat. Virtual Worlds*, 17(3-4):219–227, 2006. [1.3.1.4](#)
- [Slo03] J. J. E. Slotine. Modular stability tools for distributed computation and control. *Int. J. Adaptive Control and Signal Processing*, 17(6):397–416, 2003. [2.3](#), [1](#)
- [ST05] W. Shao and D. Terzopoulos. Artificial intelligence for animation: Autonomous pedestrians. *Proc. ACM SIGGRAPH '05*, 69(5-6):19–28, 2005. [1.3.1.2](#)
- [Str94] S. A. Strogatz. *Nonlinear Dynamics and Chaos: with applications to physics, biology, chemistry, and engineering*. Addison-Wesley Publishing Co, Reading, MA, 1994. [2](#), [2.2.2](#)
- [TCd06] M. C. Tresh, V. C. Cheung, and A. d’Avella. Matrix factorization algorithms for the identification of muscle synergies: evaluation on simulated and experimental data sets. *Journal of Neurophysiology*, 95:2199–2212, 2006. [1.3.2.1](#)
- [TCP06] A. Treuille, S. Cooper, and Z. Popović. Continuum crowds. *Proc. ACM SIGGRAPH '06*, 25(3):1160–1168, 2006. [\(document\)](#), [1.3.1.2](#), [1.3](#), [1.3.3](#), [5.2.1](#), [6](#)
- [Ter09] D. Terzopoulos. Artificial life and biomechanical simulation of humans. *Digital Human Symposium 2009*, pages 8–13, 2009. [1.3.3](#)
- [TLP07] A. Treuille, Y. Lee, and Z. Popović. Near-optimal character animation with continuous control. *Proc. SIGGRAPH '07*, 26(3), 2007. [\(document\)](#), [1.4](#), [1.3.1.3](#)

- [TPB⁺89] D. Terzopoulos, J. Platt, A. Barr, D. Zeltzer, A. Witkin, and J. Blinn. Physically-based modeling: past, present, and future. *Proc. ACM SIGGRAPH '89*, pages 191–209, 1989. [1.3.1.2](#)
- [TW90] D. Terzopoulos and K. Waters. Physically-based facial modelling, analysis, and animation. *J. Visualization and Computer Animation*, 1(2):73–80, 1990. [1.3.1.2](#), [1.3.3](#)
- [TWP03] W. Tang, T. R. Wan, and S. Patel. Real-time crowd movement on large scale terrains. *Theory and Practice of Computer Graphics*, pages 146–153, 2003. [1.3.3](#), [6](#)
- [TYS91] G. Taga, H. Yamaguchi, and Y. Shimizu. Self-organized control of bipedal locomotion by neural oscillators in unpredictable environment. *Biological Cybernetics*, 65(3):147–159, 1991. [1.3.2.2](#)
- [UT02] B. Ulicny and D. Thalmann. Towards interactive real-time crowd behavior simulation. *Computer Graphics Forum*, 21(4):767–775, 2002. [1.3.1.2](#)
- [Vap98] N. V. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998. [4.3.3](#)
- [VM01] N. Vlassis and Y. Motomura. Efficient source adaptivity in independent component analysis. *IEEE Trans. on neural networks*, 12(3):559–566, 2001. [3.2.2](#)
- [War06] W.H. Warren. The dynamics of perception and action. *Psychological Review*, 113(2):358–389, 2006. [5.2.1](#)
- [WP95] A. Witkin and Z. Popović. Motion warping. *Proc. ACM SIGGRAPH '95*, 29:105–108, 1995. [1.3.1.1](#)
- [WP05] B. Wang and M. D. Plumbley. Musical audio stream separation by non-negative matrix factorization. *In Proceedings of the DMRN Summer Conference*, 2005. [3.2.3](#)
- [WS05] W. Wang and J. J. E. Slotine. On partial contraction analysis for coupled nonlinear oscillators. *Biological Cybernetics*, 92(1):38–53, 2005. [2](#), [2.3](#), [2.3.3](#), [2.4](#), [2.4.2](#), [2.4.2](#)
- [ZH02] V. B. Zordan and J. K. Hodgins. Motion capture-driven simulations that hit and react. *Proc. ACM SIGGRAPH '2002 / Eurographics Symposium on Computer animation*, 29:89–96, 2002. [1.3.1.3](#)