Robotics and Autonomous Systems 57 (2009) 182–193

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

Timed trajectory generation using dynamical systems: Application to a Puma arm[☆]

Cristina Santos [*], Manuel Ferreira

Industrial Electronics Department, University of Minho, Guimaraes, Portugal

ARTICLE INFO

Article history:
Received 30 July 2007
Received in revised form
4 May 2008
Accepted 6 May 2008
Available online 15 May 2008

Keywords:
Nonlinear dynamical system
Timed real time trajectories generation
Online visual information
Puma arm robot

ABSTRACT

We present an attractor based dynamics that autonomously generates trajectories with stable timing (limit cycle solutions), stably adapted to changing online sensory information. Autonomous differential equations are used to formulate a dynamical layer with either stable fixed points or a stable limit cycle. A neural competitive dynamics switches between these two regimes according to sensorial context and logical conditions. The corresponding movement states are then converted by simple coordinate transformations and an inverse kinematics controller into spatial positions of a robot arm. Movement initiation and termination is entirely sensor driven. In this article, the dynamic architecture was changed in order to cope with unreliable sensor information by including this information in the vector field.

We apply this architecture to generate timed trajectories for a Puma arm which must catch a moving ball before it falls over a table, and return to a reference position thereafter. Sensory information is provided by a camera mounted on the ceiling over the robot. A flexible behavior is achieved. Flexibility means that if the sensorial context changes such that the previously generated sequence is no longer adequate, a new sequence of behaviors, depending on the point at which the changed occurred and adequate to the current situation emerges.

The evaluation results illustrate the stability and flexibility properties of the dynamical architecture as well as the robustness of the decision-making mechanism implemented.

1. Introduction

Trajectory planning has been extensively studied over the last few years, ranging from the addition of the time dimension to the robot's configuration space [10], visibility graph [27], cell decomposition [14] or neural networks [17]. There are several results for time-optimal trajectory planning [12,11].

Despite the efficient planning algorithms that have been developed and the advances in the control domain which validated dynamic, robust and adaptive control techniques, the path planning problem in autonomous robotics remains separated in theory from perception and control. This separation implies that space and time constraints on robot motion must be known beforehand with the high degree of precision typically required for non-autonomous robot operation. Such systems remain inflexible, cannot correct plans online, and thus fail both in non-static environments – such as those in which robots interact with humans – and in dynamic tasks or time-varying environments which are not highly controlled. In order to develop autonomous robot systems capable of operating in changing and uncertain environments, tight coupling of planning, sensing and execution are required.

A very relevant aspect of trajectory generation is timing, that is, the time structure of movement. This is a very critical issue in several robotic problems such as: the achievement of events in time varying environments; avoidance of moving objects; coordination of multiple robots or generating sequentially structured actions. Some approaches have addressed this issue [21], but essentially generate a single motor act in rhythmic fashion. The flexible activation of different motor acts in response to user demands or sensed environmental conditions is more difficult to achieve at the control level.

In this article, we address the timing problem considering tasks that involve sequentially structured actions, in which subsequent actions must be initiated only when previous actions have terminated or reached a particular phase. Our solution generates time structure at the level of control and its inspired from the neural mechanisms underlying movement control in animals [42] which are modeled in terms of nonlinear dynamical systems.

We generate time structure by assuming that complex movements can be generated through the sequencing of simpler movement primitives modeled as discrete and rhythmic dynamical systems. Discrete stands for goal directed movements, while

[☆] This paper was presented in part at the IEEE/RSJ 2007 International Conference on Intelligent Robots and Systems, Beijing, China, October 9–15, 2006.

[*] Corresponding author. Tel.: +351 914730808; fax: +351 253510189.
E-mail addresses: cristina@dei.uminho.pt (C. Santos), mjf@dei.uminho.pt (M. Ferreira).

0921-8890/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.robot.2008.05.002

rhythmic represents a motion stage based on amplitude and a frequency. This sequencing is controlled by another dynamical system. Trajectories are generated by activating and deactivating (sequencing) these primitives according to time intervals which depend on local sensor control and global task constraints. Movement initiation and termination is entirely sensor driven and autonomous sequence generation is stably adapted to changing unreliable online visual sensory information. These trajectories represent the temporal evolution of movement. Transformation onto spatial positions of a robotic manipulator are achieved by coordinate transformations and an inverse kinematics controller.

This movement decomposition and the chosen primitives are supported by current neurological and human motor control findings [32,41]. The present study aims at creating systems that autonomously bifurcate from single point attractors for discrete movements to limit cycles for rhythmic movements, when relatively low-level, noisy sensory information is used to initiate and steer action.

Specifically, we address the following questions. Is it possible to flexibly generate timed trajectories comprising sequence generation? Can the generated trajectories be stably and robustly implemented in a robot arm with modest computational resources? Here flexibility means that if the sensorial context changes such that the previously generated sequence is no longer appropriate, then a new sequence of behaviors suitable to the current situation emerges.

These questions are positively answered and shown in exemplary situations in which a PUMA robot arm catches a moving object launched by a human subject, and returns to a reference position thereafter. Target information is internally acquired by a visual system mounted over the robot. Although several applications could be implemented in order to test the proposed approach, with this particular application we intend to demonstrate the strength and flexibility of the approach when working in complex and unstructured environments and in situations that depend on interactions with humans. Our perspective is mainly an engineering one in the sense that we address the problem of how dynamical systems composed of stable limit cycles and fixed points can be designed to solve an engineering problem and a technological application. This is essential given that the particular task studied can be solved by conventional techniques.

The controller consists of a dynamical system composed of stable fixed points and a stable limit cycle (an Hopf oscillator). Trajectories are generated through the sequencing of these primitives, in which the limit cycle is activated over limited time intervals. This sequencing is controlled by a "neural" competitive dynamics which is build entirely around fixed points, at which only one neuron is active. Parameters of the neural dynamics express sensory and logical conditions for the activation of any particular neuron and the corresponding movement state. By controlling the timing of a limit cycle, the system performs well tasks with complex timing constraints. The online linkage to noisy sensorial information, is achieved through the coupling of these dynamical systems to time-varying sensory information [33,28].

The attractor based dynamics is based on previous work [35], in which the architecture was simulated, but the dynamic architecture is changed in order to cope with unreliable sensor information by including this information in the vector field. In [28,31], was implemented in a real vehicle and integrated with other dynamical architectures which do not explicitly parameterize timing requirements. We have also generated temporally coordinated movements among two PUMA arms [35] and among two vision-guided vehicles [29].

Results of the implementation of this controller on a real robot are shown. These results reveal the method's feasibility. The intrinsic properties of dynamical systems, enable the system to exhibit responsiveness and flexibility. The final behavior involves sequentially structured actions that depend on the environment situation and on the system state. The key point lyes in the manner how the spatio-temporal structure comes about without explicit design and despite a noisy and imperfect world. This is a major benefit of the proposed method compared to conventional techniques. However, classical conventional techniques are powerful, and have their place in many industrial applications in which the environment can be highly controlled. Therefore, the proposed solution should be faced as an extension of these classical methods and not as their replacement and, as such, be used for other robotics control tasks.

Interesting properties of the system include: (1) the possibility to include feedback loops in order to do online trajectory modulation and take external perturbations into account, such that the environmental changes adjust the dynamics of trajectory generation; (2) online modulation of the trajectories with respect to the amplitude of the rhythmic patterns, while keeping the general features of the original movements, (3) robustness to ambiguity in the environment; (4) the possibility to theoretically tune movement parameters such that it is possible to account for relationships among them; and (5) a global optimized behavior resulting from local sensor control and global task constraints. The analytically solvability and the generalization to sequence generation, are two distinguishable features of the proposed solution.

In the rest of the article, we will first review recent work on trajectory generation. Section 2 presents the proposed dynamical systems approach and discusses its intrinsic properties. Section 3 presents the technical setup for the catching application, the vision system, the controller architecture and behavioral specification. Section 4 presents the obtained results for several experiments. We conclude by presenting the conclusions and presenting future directions for the work (Section 5).

2. State of the art

In this work we present a model able to deal with the timing control of motor acts. This model is inspired on the ideas described on [15,37,4,38,32,18,34] and extends current work [28,31,33,29] (in particular [35], where a simulation study was discussed). We apply autonomous differential equations [33] to model the manner how behaviors related to locomotion are programmed in the oscillatory feedback systems of "central pattern generators" (CPGs) in the nervous systems [9,42]. Control approaches based on CPGs are widely used in robotics to achieve tasks which involve rhythmic motions such as biped and quadruped locomotion [37,15], juggling [7], drumming [19] and playing with a slinky toy [40].

There is a growing interest in nonlinear dynamical systems in the generation of timed trajectories: pattern generators for locomotion, potential field approaches for planning [7] and basis field approaches for limb movements [16].

The motivation for using these systems and oscillators in particular in engineering applications and in robotics is manifold. First, rather natural, stable and smooth complex movement patterns can arise from relatively simple sets of equations without the need to explicitly plan every movement detail. They solve timing and sequencing problems by forming flexible spatio-temporal patterns. Second, they exhibit the common features of structurally stable dynamical systems such as smooth changes under parameter variation and intrinsic robustness against small perturbations and noise. This structural stability makes it possible to fuse in input without destroying the autonomous dynamics of the system. Third, a small number of simple (scalar) parameters can control the output patterns.

Other properties are the low computation cost which is well-suited for real time; the possibility to synchronize with external signals and to add sensory feedback pathways. The dynamics of the system globally encode a task (i.e. the whole attractor landscape) with the goal state as the point attractor. Once properly designed, this property allows robustness against perturbations and provides the ability to smoothly recover from perturbations by means of coupling terms in the dynamics, in contrast to other approaches such as finite state machines [3]. External coupling terms can be inserted into the dynamical systems that can modulate the output of the equations in useful ways, e.g., as needed in synchronization with external events, in situations with contact forces, or in general when external perturbations need to be compensated for. In this project, the online linkage to sensory information is achieved though the coupling of the nonlinear dynamical systems to time-varying sensory information. This property enables us to include feedback loops in order to do online trajectory modulation and take external perturbations into account, such that the environmental changes adjust the dynamics of trajectory generation. Another particularity is that these systems produce coordinated multidimensional rhythms of motor activity, under the control of simple input signals. Such systems are deemed to strongly reduce the dimensionality of the control problem: only constant inputs to modulate high-dimensional oscillator outputs (in contrast to filter banks [39] or the like). Furthermore, planning in terms of autonomous nonlinear attractor landscapes promises more general movement behaviors than traditional approaches using time-indexed trajectory planning. By removing the explicit time dependency one can avoid complicated ‘clocking’ and ‘reset clock’ mechanisms.

A motivation for using the dynamical systems framework in the context presented in this article is the possibility to flexibly generate structured actions according to the current sensorial context and the system state. The proposed solution creates spatio-temporal patterns in which the observed, emergent patterns are described by a small number of variables. These patterns can be modulated by some parameters, which offers the possibility to smoothly modulate the trajectories.

Furthermore, the proposed approach is based on a rigorous mathematical framework using nonlinear dynamical systems. In robotics, in most approaches, the proposed solutions are not based on a rigorous framework, i.e. they are based on heuristics and ad-hoc approaches. Since the design of the model is entirely described by continuous nonlinear differential equations on continuous behavioral variables, we guarantee the stability and the controllability of the overall system by obeying the time scale separation principle. This also leads to a continuous behavior.

Further, typically engineering approaches decompose trajectory planning and control in which the application of the proposed controller on a robot would be on top of a rather classical level of low level control that ensures that the trajectories are followed. Here we argue that to not fully control every aspect of the robot provides for higher flexibility.

In this work, decision making, timed path planning and control are generated from attractor solutions of nonlinear dynamical systems. The possibility of integrating multiple constraints and generating decisions through instabilities and multistability makes such systems much more flexible than other more conventional techniques. We take benefit of intrinsic properties, such as stability, bifurcation, and hysteresis, that enable planning decisions to be made and carried out in a flexible, yet stable, way even if noisy sensory information is used to initiate action. Another possible approach would be by means of symbolic computation. However, in such case one looses flexibility, an intrinsic property of the dynamic approach, achieved in that planning decisions may change continuously, but also discontinuously in response to the changing sensed environment. Changes in sensory information lead to a qualitative change of behavior brought about by instabilities. Decision making is designed by using bifurcation theory such that the behavioral dynamics go through a specific and chosen bifurcation under the adequate conditions. Conversely, symbolic computation requires that every possible behavioral situation is known in advance such that an algorithm defines which behavior must be active for every combination of possible behavioral states.

The hysteresis property of the dynamic approach leads to a stable integration of decision making into the control behavior and path planning. This property enables the maintenance of stability within the decision zone in ambiguous situations. The system is prevented from oscillating among possible solutions due to hysteresis in case of multistable dynamics. By contrast, in symbolic computation the switch among possible behaviors would happen instantaneously possibly violating the requirement that the system should at all times be in or near to an attractor state of the dynamical system. Thus, the missing stability property of the symbolic computation makes it difficult to integrate into stable control.

Furthermore, the dynamic approach can easily be scaled up to the design of more complex tasks. By contrast, the integration of new behavioral requirements within a symbolic algorithm requires the reconfiguration of the overall structure of the previous system.

Within our approach there is no differentiation between logics and control since the logics is contained in the parameters of the differential equations and not in an explicit program. A smooth stable integration of discrete events and continuous processes is thus achieved. The inherent stability property of the dynamic approach to control is carried over to logic control.

Another general advantage of this approach is that it does not make unreasonable assumptions, or place unreasonable constraints on the environment in which the robot operates. The fact that the dynamical architecture does not explicit represent any properties of the real world and establishes a direct linkage between actuators and sensors assures a quick reaction to eventual changes in the sensed environment. This characteristic is shared by some other approaches in the field of artificial life [24,6].

To the best of our knowledge, the framework of sequencing of discrete and rhythmic movements has not been applied to timed movement generation within robotics (but see [32,18]). Other solutions [26,21,8,40] have tried to address the timing problem, by generating time structure at the level of control. More generally, the nonlinear control approach to locomotion pioneered by [26] amounts to using limit cycle attractors that emerge from the coupling of a nonlinear dynamical control system with the physical environment of the robot. A limitation of such approaches is that they essentially generate a single motor act in rhythmic fashion, and remain limited with respect to the integration of multiple constraints, and planning was not performed in the fuller sense.

However, as it was stated previously, our perspective is mainly an engineering one in the sense that we attempted to solve an engineer problem and a technical application using dynamical systems. It also differs from most of the literature in that it is implemented on a real robot and extends the use of oscillators to tasks on an arm robot. In alternative engineering approaches, e.g. visual servoing, very powerful results and methods have been developed which should be used as long as appropriate.

3. Timed trajectories generation for a PUMA arm

In this article we try to solve a robotic problem applying the dynamical systems approach to timing. Fig. 1 depicts the problem setup: a PUMA arm must catch a green ball at the end of the

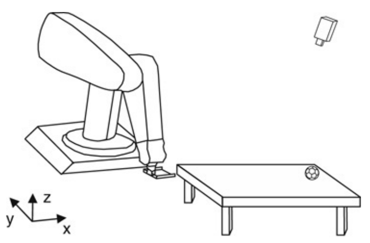


Fig. 1. The PUMA arm must catch the ball before it falls over the table. The x and z coordinates for catching position are fixed and known. A camera acquires visual information that enables the system to calculate the y coordinate of catching position and the time it takes for the ball to be at that location.

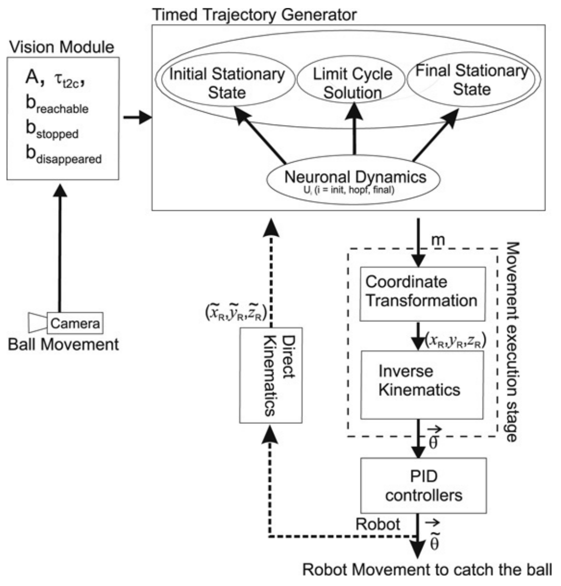


Fig. 2. The overall architecture of the system. Ball movement is detected by a visual system and transformed onto time-varying parameters of a timed trajectory controller. Timed movement in real time, m, is obtained by integrating the dynamical systems. An additional movement execution stage by means of coordinate transformations and an inverse kinematics controller transforms this timed movement onto planned joint values $\vec{\theta}$. Real joint values, $\vec{\tilde{\theta}}$, result from the servo motors of the PUMA.

table on which the ball is moving. The task is to generate a timed movement from an initial posture to intercept an approaching ball. Movement with a fixed movement time (reflecting manipulator constraints) must be initiated in time to catch the ball before it falls over the table. Factors such as reachability and approach path of the ball are continuously monitored through online visual sensory information, leading to a return of the arm to the resting position when catching becomes impossible (e.g., because the ball hits outside the workspace of the arm, the ball is no longer visible, or ball contact is no longer expected within a criterion time-to-contact). After catching the ball, the arm moves back to its resting position, ready to initiate a new movement whenever appropriate sensory information arrives.

3.1. Overall architecture

The overall system architecture is depicted in Fig. 2. The green ball movement is captured by a vision module which calculates the point-of-contact ((x, y, z) coordinates of catching position) and the time-to-contact, τ_{t2c}, that is, the time it takes to the ball to intersect the arm at this point in space.

Timed movement in real time is generated by a controller formulated in terms of nonlinear dynamical systems for dynamical variables (m, n). Timed movement is obtained by integrating these dynamical systems. The time courses of the m dynamical variable evolve from an initial to a final value, yielding a timed movement with amplitude A. The state of the movement is represented by the dynamical variable, m, which is not directly related to the spatial position of the PUMA end-effector, but rather represents the effector temporal position.

Ball movement is transformed onto time-varying parameters that control the parameters of the (m, n) dynamical systems that generate timed movement in real time. More precisely, these parameters specify the time-to-contact, τ_{t2c}, and the amplitude A of the movement, controlled by the point-of-contact given by the vision module. The inclusion of these feedback-loops enables online trajectory modulation.

PUMA end-effector and ball movement are expressed in a world cartesian reference frame. PUMA end-effector position varies along a straight path from the initial PUMA end-effector resting position to the ball catching position. Transformation from the temporal dynamical variable, m, to PUMA end-effector spatial position, (x_R, y_R, z_R), is achieved by simple coordinate transformations (see Eq. (13)) that depend on the calculated ball catching position, point-of-contact, which is given by the vision module.

An additional movement execution stage by means of an inverse kinematics controller (based on the geometrical solution [13]) transforms these desired positions onto motor commands which are used as inputs for the servo motors of the PUMA robot and result in the actual $\vec{\tilde{\theta}}$ joint values and actual $(\tilde{x}_R, \tilde{y}_R, \tilde{z}_R)$ PUMA end-effector positions.

A starting end-effector orientation is established and kept constant during motion. During movement execution, the dynamical variable, m, is continuously transformed into PUMA end-effector position, (x_R, y_R, z_R), from which joint angles, $\vec{\theta}$, are computed through the inverse kinematic transformation.

3.2. The dynamical systems trajectory generator

Our aim is to propose a control architecture that is able to generate trajectories for a robot arm such that it reaches in time an approaching ball. These trajectories should be smoothly modulated when simple control parameters change.

The proposed controller is modeled by a dynamical system that can generate trajectories that have both discrete and rhythmic components. The system starts at an initial time in an initial discrete position, and moves to a new final discrete position, within a desired movement time, and keeping that time stable under variable conditions. The final discrete position and movement initiation change and depend on the visually detected ball, on the current joint values and on the robot internal model. Thus, trajectories generated by this architecture are modulated by sensory feedback.

The overall controller architecture is depicted in Fig. 3.

3.2.1. Fixed points and limit cycle solutions generator

The developed controller is divided in three subsystems, one generating the initial discrete part of movement, another generating the oscillatory part and another generating the final discrete part of movement. A dynamical system for a pair of behavioral variables (m, n) is defined to generate the timed movement [35,28]. Although only the variable, m, will be used to set the robotic variable, a second auxiliary variable, n, is needed to enable the system to undergo periodic motion.

This dynamical system can operate in three dynamic regimes that correspond to the stable solutions of the individual dynamical

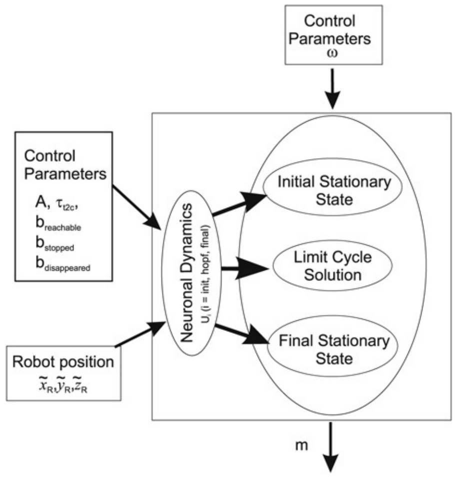


Fig. 3. Mechanism for timed trajectory generation. Timed movement is generated through the sequencing of stable fixed points and a stable limit cycle. This sequencing is controlled by a neural competitive dynamics according to sensorial context and logical conditions. Trajectories are modulated by particular choices of parameter *A*.

systems: two discrete states (stationary states) and a stable oscillation (a limit cycle solution). It is defined as follows:

$$\begin{pmatrix} \dot{m} \\ \dot{n} \end{pmatrix} = 5|u_{\text{init}}| \begin{pmatrix} m \\ n \end{pmatrix} + |u_{\text{hopf}}| f_{\text{hopf}} + 5|u_{\text{final}}| \begin{pmatrix} m - A \\ n \end{pmatrix} + \text{gwn}. \tag{1}$$

The "init" and "final" contributions describe a discrete motion whose solutions converge asymptotically to a globally attractive point at $m = 0$ for "init" and $m = A$ for "final" with $n = 0$ for both. Speed of convergence is controlled by $\tau = 1/5 = 0.2$ time units. If only the final contribution is active ($u_{\text{hopf}} = u_{\text{init}} = 0$; $|u_{\text{final}}| = 1$), each time A is changed, the system will be attracted by the new A value, generating a discrete movement towards A.

The "Hopf" term describes an Hopf oscillator, that generates the limit cycle solution (as defined in [35,28]) and is given by:

$$\mathbf{f}_{\text{hopf}} = \begin{pmatrix} \alpha & -\omega \\ \omega & \alpha \end{pmatrix} \begin{pmatrix} \left(m - \frac{A}{2}\right) \\ n \end{pmatrix} - \gamma \left(\left(m - \frac{A}{2}\right)^2 + n^2 \right) \begin{pmatrix} \left(m - \frac{A}{2}\right) \\ n \end{pmatrix} \tag{2}$$

where $\gamma = \frac{4\alpha}{A^2}$ controls the amplitude of the oscillations, ω is the oscillator intrinsic frequency and α controls the speed of convergence to the limit cycle. This oscillator in isolation ($u_{\text{init}} = u_{\text{final}} = 0$; $|u_{\text{hopf}}| = 1$), contains a bifurcation from a fixed point (when $\alpha < 0$) to a structurally stable, harmonic limit cycle with radius $A = 2\sqrt{\frac{\alpha}{\gamma}}$, cycle time $T = \frac{2\pi}{\omega} = 4$ time units and relaxation to the limit cycle given by $\frac{1}{2\alpha\gamma}$, for $\alpha > 0$. Thus, it provides a stable periodic solution (limit cycle attractor)

$$m(t) = \frac{A}{2} + \frac{A}{2}\sin(\omega t). \tag{3}$$

Because the system is analytically treatable to a large extent, it facilitates the specification of parameters such as frequency, amplitude or offset. This analytical specification is an innovative aspect of our work. Relaxation to the limit cycle solution occurs at a time scale of $1/(2\alpha) = 0.2$ time units.

The fixed point m has an offset given by $\frac{A}{2}$. For $\alpha < 0$ the system exhibits a stable fixed point at $m = \frac{A}{2}$.

This Hopf oscillator describes a rhythmic motion which amplitude of movement is specified by A and its frequency by ω.

The dynamics of (1) are augmented by a Gaussian white noise term, gwn, that guarantees escape from unstable states and assures robustness to the system.

The switching between the 3 possible modes of movement is controlled by a neural dynamics through three "neurons" u_i ($i =$ init, hopf, final). This switch is controlled by several parameters including calculated time-to-contact and point-of-contact, acquired by the vision system. Moreover, the amplitude A of movement depends on the calculated point-of-contact and this provides for online trajectory modulation. By modifying on the fly these parameters, one can easily generate different stable trajectories.

Here the system is able to cope with fluctuations in amplitude A because quantities that depend on sensory information are included in the vector field. Online trajectory modulation is achieved through the inclusion of this feedback loop that enables us to take ball movement into account, such that when a change occurs in the ball movement, the system online adjusts the dynamics of trajectory generation.

3.2.2. Neural dynamics

The "neuronal" dynamics of $u_i \in [-1, 1]$ ($i =$ init, final, hopf) switches the dynamics from the initial and final stationary states into the oscillatory regime and back. Thus, a single discrete movement act is generated by starting out with neuron $|u_{\text{init}}| = 1$ activated, the other neurons deactivated ($|u_{\text{hopf}}| = |u_{\text{final}}| = 0$), so that the system is in the initial stationary state ($m = 0$). Then, neuron $|u_{\text{init}}| = 0$ is deactivated and neuron $|u_{\text{hopf}}| = 1$ activated and the system evolves along the oscillatory solution. After approximately a half-cycle of the oscillation, this oscillatory solution is deactivated again turning on the final postural state instead ($|u_{\text{hopf}}| = 0$; $|u_{\text{final}}| = 1$).

These switches are controlled by the following competitive dynamics

$$\alpha_u \, \dot{u}_i = \mu_i \, u_i - |\mu_i| \, u_i^3 - 2.1 \sum_{a,b \neq i} \left(u_a^2 + u_b^2\right) u_i + \text{gwn}, \tag{4}$$

where "neurons", u_i, can go "on" (=1) or "off" (=0). The first two terms of the equation represent the normal form of a degenerate pitchfork bifurcation: A single attractor at $u_i = 0$ for negative μ_i becomes unstable for positive μ_i, and two new attractors at $u_i = 1$ and $u_i = -1$ form. We use the absolute value of u_i as a weight factor in (1).

The third term is a competitive term, which destabilizes any attractors in which more than one neuron is "on". For positive μ_i, all attractors of this competitive dynamics have one neuron in an "on" state, and the other two neurons in the "off" state [34,20]. The dynamics of (4) are augmented by the Gaussian white noise term, gwn, that guarantees escape from unstable states and assures robustness to the system.

Fig. 4 presents a schematic illustrating this dynamics. This dynamics enforces competition among task constraints depending on the neural competitive parameters ("competitive advantages"), μ_i. As the environmental situation changes, the competitive parameters reflect by design these changes causing bifurcations in the competitive dynamics. The neuron, u_i, with the largest competitive advantage, $\mu_i > 0$, is likely to win the competition, although for sufficiently small differences between the different μ_i values multiple outcomes are possible (the system is multistable) [20].

In order to control switching, μ_i parameters are explicitly designed such that their functions reflect the current sensorial context and the global constraints expressing which states are

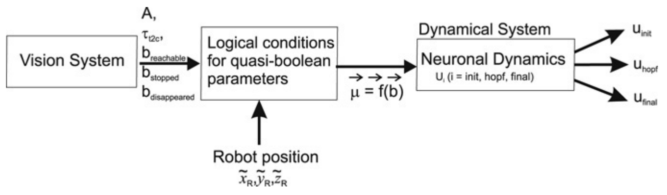

Vision System
A, τt2c, breachable, bstopped, bdisappeared
Logical conditions for quasi-boolean parameters
μ = f(b)
Dynamical System
Neuronal Dynamics
Ui (i = init, hopf, final)
uinit
uhopf
ufinal
Robot position
x̃R, ỹR, z̃R

Fig. 4. Schematic representation of the neural dynamics. Current sensorial context and global constraints change as the environmental situation changes. By design, the μ_i parameters reflect these changes causing bifurcations in the neural dynamics and activation of a neuron $u_i \in [-1, 1]$ (i = init, final, hopf). These neurons enable the system to appropriately control sequencing of movement primitives.

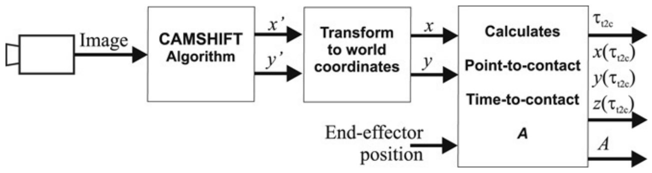


Fig. 5. Vision module.

more applicable to the current situation. They are defined as functions of robot position, parameters returned by the visual system and internal states and control the sequential activation of the different neurons (see [36], for a general framework for sequence generation based on these ideas and [35] for a description). Herein, we vary the μ-parameters between the values 1.5 and 3.5: $\mu_i = 1.5 + 2b_i$, where b_i are "quasi-boolean" factors taking on values between 0 and 1 (with a tendency to have values either close to 0 or close to 1). Hence, we assure that one neuron is always "on".

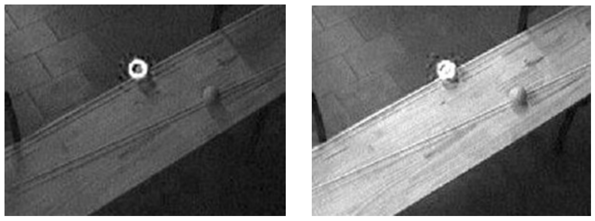
The time scale of the neuronal dynamics is set to a relaxation time of $\tau_u = \frac{1}{\alpha_u} = 0.02$, ten times faster than the relaxation time of the (m, n) dynamical variables. This difference in time scale guarantees that the analysis of the attractor structure of the neural dynamics is unaffected by the dependence of its parameters, μ_i on the dynamical variable, m, which is a dynamical variable as well. Strictly speaking, the neural and timing dynamics are thus mutually coupled. The difference in time scale makes it possible to treat m as a parameter in the neural dynamics (adiabatic variables). Conversely, the neural weights can be assumed to have relaxed to their corresponding fixed points when analyzing the timing dynamics (adiabatic elimination). The adiabatic elimination of fast behavioral variables reduces the complexity of a complicated behavioral system built up by coupling many dynamical systems [31,36]. By using different time scales one can design the several dynamical systems separately.

3.3. Coupling to sensorial information

In order to intercept an approaching ball it is necessary to be at the right location at the right time. We use the visual stimulus as the perception channel to our system by mounting a camera in the ceiling over the robot workspace.

In our application, the goal is to robustly track a green ball moving in a table in an unstructured, complex environment, using inexpensive consumer cameras and avoiding calibration lenses procedures. Specifically, we have to deal with the following main computer-vision problems: (1) a clutter environment, including non-uniform light conditions and different objects with the same color pattern (distractors); (2) irregular object motion due to perspective-induced motion irregularities; (3) image noise and (4) a real-time performance application with high processing time.

The overall vision module showing outputs and information flow is depicted in Fig. 5.



(a) Presence of a distractor element. (b) Variations in lighting conditions.

Fig. 6. Application of the CAMSHIFT algorithm to a real, clutter environment, where some computer vision problems in visual object tracking are addressed.

The most common algorithms for visual object tracking in robot applications are typically based on the detection of a particular cue, most commonly edges, color and texture [23,25,2,5,22,1]. Conventional single-cue algorithms typically fail outside limit tracking conditions which degrade performance. Although such algorithms fail to catch variations like changes of orientation and in shape, if flexibility and/or simplicity, speed and robustness are required, as in our case, they are a good option. Specifically, we have chosen a color based real-time tracker, Continuously Adaptive Mean Shift (CAMSHIFT) algorithm [5], that handles the described computer-vision application problems during its operation. This algorithm uses a search window to track the moving object, ignoring objects outside this search window. Further, handles perspective-induced motion irregularities by scaling the search window to object size. The used color space is the HSV which eliminates much of the noise present in the image. Using only the hue histogram and ignoring pixels with high or low brightness provides wide lighting tolerance.

The CAMSHIFT algorithm segments the image and tracks the x', y' image coordinates and area of the color blob representing the green ball.

Image coordinates are transformed onto world coordinates by a general perspective transformation assuming that the table height is fixed and known (the Z coordinate remains the same for the entire ball path) and that camera lens with a focal distance which produces negligible distortion have been used.

We illustrate two real applications of this algorithm to a real, clutter environment where the environment has non-uniform light conditions and there exist several different objects with the same color pattern. Fig. 6 shows the result of this algorithm in the presence of a distractor element. In Fig. 6 the incident illumination as been increased by a factor of 1.5. In both situations, the algorithm is able to track the ball.

In this application, the robot arm catches the ball at the end of the table in which the ball is moving (Fig. 1). This is point-of-contact $(x(\tau_{t2c}), y(\tau_{t2c}), z(\tau_{t2c}))$.

For simplicity, in this application $x(\tau_{t2c})$ is always constant and has a known value ($x(\tau_{t2c}) = -154$ mm). $z(\tau_{t2c})$ corresponds to the table's height and thus $z(\tau_{t2c}) = -519$ mm. Further, we consider that ball movement has a linear trajectory in the 3D cartesian space with a constant approach velocity. Thus, time-to-contact, τ_{t2c}, and $y(\tau_{t2c})$ are extracted from the obtained visual information through straightforward formulae:

$$y(\tau_{t2c}) = \frac{(x(\tau_{t2c}) - x(0))\ (y(0) - y(t))}{x(0) - x(t)} + y(0) \tag{5}$$

$$\tau_{t2c} = \frac{\sqrt{(x(\tau_{t2c}) - x(t))^2 + (y(\tau_{t2c}) - y(t))^2}}{\sqrt{v_x^2 + v_y^2}} \tag{6}$$

where $x(0)$ and $y(0)$ are the coordinates of the ball at $t = 0$; v_x and v_y the velocity of the ball as calculated by the CAMSHIFT algorithm.

Movement amplitude, A, is set as,

$$A = \sqrt{(y(\tau_{t2c}) - y_R(0))^2 + (z(\tau_{t2c}) - z_R(0))^2}, \tag{7}$$

where $y_R(0)$ and $z_R(0)$ denote end-effector position previously to movement initiation.

3.4. Behavior specifications

These two measures, time-to-contact and point-to-contact, fully control the neural dynamics through the quasi-boolean parameters. A sequence of neural switches is generated by translating sensory conditions and logical constraints into values for these parameters.

The parameter, b_{init}, controlling the competitive advantage of the initial postural state, is controlled by sensory input: it changes from 1 to 0 when the time-to-contact of the approaching ball computed from sensory information is below a certain value. Movement is initiated in this manner. b_{init} must be "on" $(=1)$ when the sensed actual position of the effector is close to the initial state 0 ($b_{x_R \text{ close } x_{\text{init}}}(x)$); **and** either of the following is true: (1) ball not approaching or not visible ($\tau_{t2c} \leq 0$); (2) ball contact not yet within a criterion time-to-contact ($\tau_{t2c} > \tau_{\text{crit}}$); (3) ball is approaching within criterion time-to-contact but is not reachable ($0 < \tau_{t2c} < \tau_{\text{crit}}$; $b_{\text{reachable}} = 0$); (4) ball stopped ($b_{\text{stopped}} = 1$); (5) ball was caught ($b_{\text{caught}} = 1$); (6) ball has disappeared ($b_{\text{disappeared}} = 1$).

Note that sensed actual position of the effector, x_{robot}, represents the temporal position of the effector. It has been transformed from the spatial position of the effector in world coordinates, $(\tilde{x}_R, \tilde{y}_R, \tilde{z}_R)$.

The factor $b_{x_R \text{close} x_{\text{init}}}(x_{\text{robot}}) = \sigma(0.15A - x_{\text{robot}})$ has values close to one while the sensed actual position of the effector is bellow $0.15A$ and switches to values close to zero elsewhere. This switch is driven from the sensed actual position of the robot. Herein, $\sigma(\cdot)$ is a sigmoid function that ranges from 0 for negative argument to 1 for positive argument, chosen here as

$$\sigma(x) = [\tanh(100x) + 1]/2, \tag{8}$$

although any other functional form will work as well.

These logical conditions can be expressed through this mathematical function:

$$\begin{aligned} b_{\text{init}} = {} & \sigma(0.15A - x_{\text{robot}})[\sigma(\tau_{t2c} - \tau_{\text{crit}}) + \sigma(-\tau_{t2c}) \\ & + \sigma(\tau_{t2c})\,\sigma(\tau_{\text{crit}} - \tau_{t2c})\,\sigma(1 - b_{\text{reachable}}) \\ & + b_{\text{stopped}} + b_{\text{caught}} + b_{\text{disappeared}}]. \end{aligned} \tag{9}$$

(b_{stopped}), b_{caught} and $b_{\text{disappeared}}$ are flags set to 1 in order to indicate that the ball stopped, was caught or disappeared, respectively. These conclusions are taken dependent on the acquired visual sensory information.

Multiplication and sum of "quasi-booleans" realizes the "and" and "or" operations among logical conditions, respectively.

A similar analysis derives the b_{hopf} parameter, controlling the competitive advantage of the oscillatory state. b_{hopf} parameter must be "on" $(=1)$ when either of the following is true:

1. The sensed actual position of the effector is no longer close to the initial state 0 ($\sigma\,(x_{\text{robot}} + 0.15A)$), **and** either of the following is true: (1) ball contact not yet within a criterion time-to-contact ($\tau_{t2c} > \tau_{\text{crit}}$); (2) ball not approaching or not visible ($\tau_{t2c} < 0$); (3) impact is not reachable ($b_{\text{reachable}} = 0$); (4) the sensed actual position of the effector is not yet close to the final postural state A ($\sigma\,(0.85A - x_{\text{robot}})$); (5) ball stopped ($b_{\text{stopped}} = 1$); (6) ball was caught ($b_{\text{caught}} = 1$); (7) ball has disappeared ($b_{\text{disappeared}} = 1$).
2. Ball is approaching within criterion time-to-contact ($0 < \tau_{t2c} < \tau_{\text{crit}}$); **and** point of impact is reachable ($b_{\text{reachable}} = 1$); **and** the sensed actual position of the effector is not yet close to the final postural state A ($\sigma\,(0.85A - x_{\text{robot}})$).

In the following expression for b_{hopf}, the "or" is expressed with the help of the "not" (subtracting from 1) and the "and":

$$\begin{aligned} b_{\text{hopf}} = {} & 1 - (1 - [\sigma(0.15A - x_{\text{robot}})\,\sigma(\tau_{t2c})\sigma(\tau_{\text{crit}} - \tau_{t2c}) \\ & \times \sigma(b_{\text{reachable}})]) \cdot (1 - [\sigma(x_{\text{robot}} + 0.85A) \\ & \times \{\sigma(1 - b_{\text{reachable}}) + \sigma(-\tau_{t2c}) + \sigma(\tau_{t2c} - \tau_{\text{crit}}) \\ & + \sigma(0.85A - x_{\text{robot}}) + \sigma(b_{\text{stopped}}) + \sigma(b_{\text{caught}}) \\ & + \sigma(b_{\text{disappeared}})\}]). \end{aligned} \tag{10}$$

Analogously, b_{final}, which controls the competitive advantage of the final postural state, must be "on" $(=1)$ when none of the following is false: (1) ball is approaching within criterion time-to-contact ($0 < \tau_{t2c} < \tau_{\text{crit}}$) **and**; (2) estimated impact is reachable ($b_{\text{reachable}}$) **and**; (3) the sensed actual position of the effector is close to the final postural state A. This parameter is given by:

$$\begin{aligned} b_{\text{final}} = {} & \sigma(\tau_{t2c})\sigma(\tau_{\text{crit}} - \tau_{t2c})\sigma(b_{\text{reachable}})\sigma(x_{\text{robot}} - 0.85A) \\ & \times \sigma(1 - b_{\text{stopped}})\sigma(1 - b_{\text{disappeared}}). \end{aligned} \tag{11}$$

3.5. Velocity

The puma velocity, V_{puma}, is set as:

$$V_{\text{puma}} = |\dot{m}|, \tag{12}$$

where m is the dynamical variable given by (1).

Previously to timed trajectory generation, the robot arm is moved to a pre-defined location, $(x_R, y_R, z_R)(0) = (-154, -393, -123)$ mm. Robot arm movement only happens in the Z and Y plane because $x_R(0)$ equals $x(\tau_{t2c})$. The m dynamical variable is then transformed onto the y_R and z_R coordinates of robot movement by the following coordinate transformations

$$\begin{aligned} x_R &= x_R(0) \\ y_R &= y_R(0) + \cos(\beta)\, m \\ z_R &= z_R(0) + \sin(\beta)\, m, \end{aligned} \tag{13}$$

where

$$\beta = \arctan \frac{z(\tau_{t2c}) - z_R(0)}{y(\tau_{t2c}) - y_R(0)}. \tag{14}$$

3.6. Robotic setup

The dynamic architecture was implemented and evaluated on a PUMA arm. A schematic of the implementation is presented in Fig. 7. The PUMA 560 is a six-joint industrial robot manipulator, whose original LSI/11 processor, the VAL-II operating system and the joint controllers, were replaced by a new system based on the Trident Robotics cards: TRC004, TRC100, TRC041 (Puma cable card set) [30] and a personal computer (PC). The TRC004 is a general purpose interface board for servo applications, mounted and wired to the backplane by the TRC041 card. The TRC100 is a general purpose RISC processor board for servo control and data acquisition applications. In our case, this card provides an interface between the TRC004 and the ISA bus of the PC. The interface between the TRC100 and TRC004 is accomplished by a software developed by Trident Robotics (a DLL for Windows environment and some software placed in the EPROM of TRC100). This new installed architecture gives direct access to the joint positions and bypasses the old joint controllers, enabling the implementation of new strategies for each of the joint controllers, the generation of trajectories or task planning algorithms. The PUMA arm is set in a blocking mode. In order to process and start a movement command the PUMA controller takes around 40 ms.

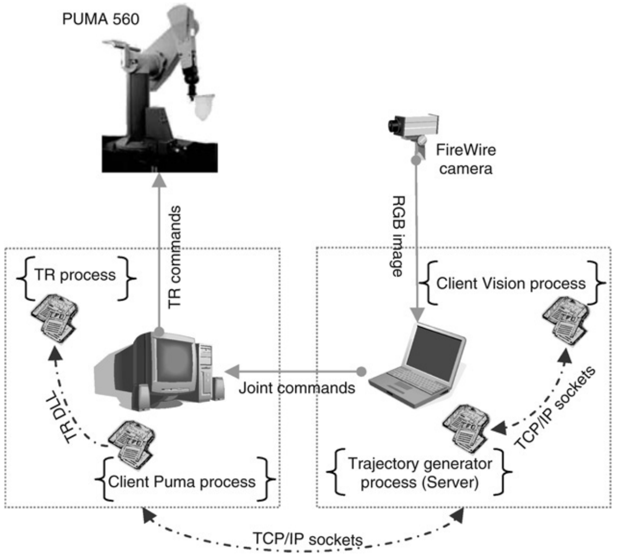


Fig. 7. Schematic depicting the robotic setup. TR stands for Tridents Robotics.

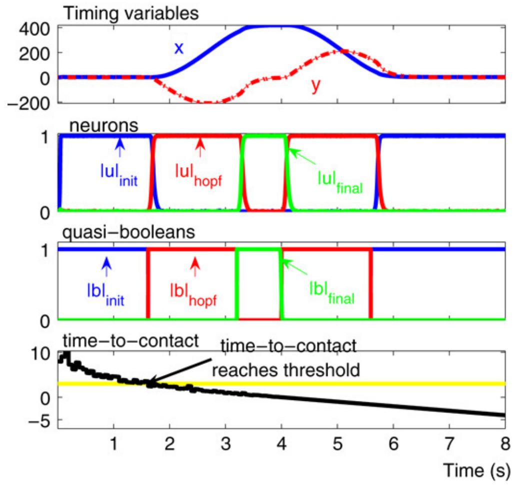


Fig. 9. Trajectories of variables and parameters in autonomous ball catching and return to resting position. The top three panels represent timing variables, neural variables and quasi-booleans. The bottom panel shows the time-to-contact, which crosses a threshold at about 3 time units. At this moment, the arm initiates its timed movement.

The CCD color cameras are FireWire with a resolution of 640 × 480 pixels RGB. Image processing is done on a 2GHz Pentium M PC. The image sensorial cycle for acquiring and processing an image takes around 63 ms.

During tracking, the processing time of the CAMSHIFT algorithm has a mean value of 14.3 ms and a standard deviation of 0.7 ms. When the algorithm, during the ball movement, looses the track of the ball the search window is resized to full image size. The processing time increases to a mean value of 17.8 ms with a standard deviation of 0.9 ms.

The dynamics of timing and competitive neural of the trajectory generator are numerically integrated in this PC using the Euler method with a Euler step around 2 ms (cycle time or sensorial cycle).

The two PCs are connected to an Ethernet network. In order to exchange information between the three processes, we implemented a communication mechanism based on sockets. This interprocess communication uses the client server model, in which the trajectory generator process (the server), connects to the vision and puma processes (the clients) to make a request for information. By applying this process separation, we obtain independent processes and may consider that the cycle time for the trajectory generator is 2 ms if PUMA position is updated only every 20 sensorial cycles and considering an image is acquired only every 35 sensorial cycles. This yields a movement time (MT) of 2 s.

4. Experimental results

In order to ensure the generality of the proposed architecture we have performed several experiments. Frequency has been set to $\omega = \frac{\pi}{MT}$ in regards with motor and other hardware limitations, but keeping the demonstration feasible. Dynamical parameters controlling speed of convergence of each dynamical system were chosen in order to respect stability during the integration process; the required separation in the time scales and feasibility of the experiments, as previously described. Trajectories from the joints incremental encoders were recorded as well as planned trajectories.

4.1. A simple experiment

The sequence of video images shown in Fig. 8 illustrates the robot motion when the PUMA arm successfully catches an approaching ball. In the video sequence, time increases from left to right.

The detailed time courses of the relevant variables and parameters are shown in Fig. 9. The real point-to-contact is at $(x, y, z) = (-154, -553, -519)$ mm which yields a movement amplitude of 427 mm.

As the ball approaches, the current time-to-contact becomes smaller than a critical value (here 3 s), at which time the quasi-boolean for motion, b_{hopf} becomes one, triggering activation of the corresponding neuron, u_{hopf}, and movement initiation. Movement is completed (m dynamical variable varies between the initial postural state at zero and the final postural state at $A = 422$ mm) before actual ball catching is made. The arm waits in the final posture. Ball catching is detected by the vision system which activates the b_{caught} flag and leads to autonomous initiation of the backward movement to the arm resting position.

Fig. 10 depicts snapshots of the PUMA arm for the situation where the point-of-contact is unreachable. Movement is not initiated and the arm rests in its reference position.

A rate of failure of 12% is achieved when 20 experiments are done for the same ball movement. Let $d_{\text{collision}}$ represent the distance between the end-effector position and the real ball position at the point-to-contact location. The mean value of this

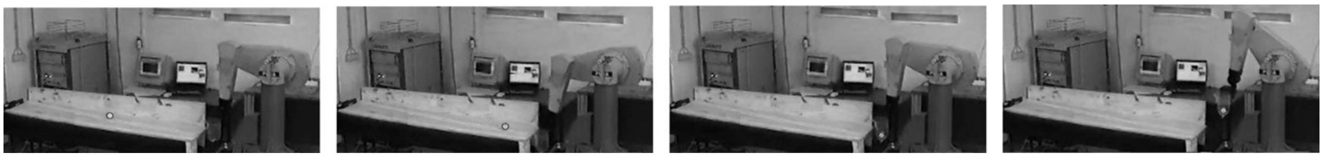


Fig. 8. A sequence of video images illustrates robot motion when the PUMA arm successfully catches an approaching ball (left to right). Point-to-contact is at $(x, y, z) = (-154, -553, -519)$ mm which yields a movement amplitude of 427 mm. Robot movement is autonomously initiated at $t = 3$ s. Movement is completed before actual ball catching is made. Ball catching is successfully detected by the vision system and an autonomous initiation of the arm back to the resting position is done.

Fig. 10. A sequence of video images illustrates system behavior when the approaching ball is unreachable.

variable within the 20 simulations is 4.5 mm, while in cases where no solution was proposed to cope with a noisy point-to-contact (simulation work described in [35]), the mean value of $d_{\text{collision}}$ was 100 mm. The proposed solution leads to improvement.

4.2. Properties of the generated timed trajectory

An advantage of this approach is that because the system is analytically treatable, smooth trajectory online modulation of the trajectories with respect to the amplitude and frequency parameters is now possible, while keeping the general features of the original movements. A simple modulation of the parameters can generate an infinite variation of stable trajectories.

Different points-of-contact calculated using visual online sensory information result in different reaching trajectories which are generated in real-time. Trajectories are thus modulated according to the environment, such that action is steered by online modulation of the parameters. Meaning, there is no need to re-design the system for different points-of-contact and time-to-contact.

The system is conceived to catch any point-of-contact within the reachable workspace and several ones were tried. Here in, we only present results for the one depicted in Fig. 8, as results were basically the same for different locations.

The fact that trajectories are generated online by integrating differential equations in real time, allows to change the parameters on the fly. For instance, if the ball is abruptly moved, a new point-of-contact is calculated and the system smoothly changes the trajectories accordingly. These feedback loops enable us to take ball movement into account, and this provides for online trajectory modulation. This situation is illustrated in Fig. 15.

4.2.1. Stabilization of decision to initiate movement

Intrinsic stability properties are inherent to the Hopf oscillator, which has a structurally stable limit cycle. Thus, the generated trajectories are robust to the presence of noise and stable to perturbations.

The y_R and z_R real robot trajectories for experiment shown in Fig. 9 are illustrated in Fig. 11. Noisy sensory information produces amplitude fluctuations in point-to-contact. However, these fluctuations are included in the vector field and thus are filtered. Despite the noisy amplitude, the robot trajectories are almost not affected.

This experiment illustrates how the generation of the timing sequence resists against sensor noise: the noisy time and point-to-contact data led to strongly fluctuating quasi-booleans (noise being amplified by the threshold functions). The neural and timing dynamics, by contrast, are not strongly affected by sensor noise so that the timing sequence is performed as required. This demonstrates approach robustness. This property is specially useful for adding feedback pathways because sensory information is forgotten as soon as it disappears from the environment. Note how the autonomous sensor-driven initiation of movement is stabilized by the intrinsic hysteresis properties of the competitive neural dynamics, so that small fluctuations of the input signal back above threshold do not stop the movement once it has been initiated [34,35].

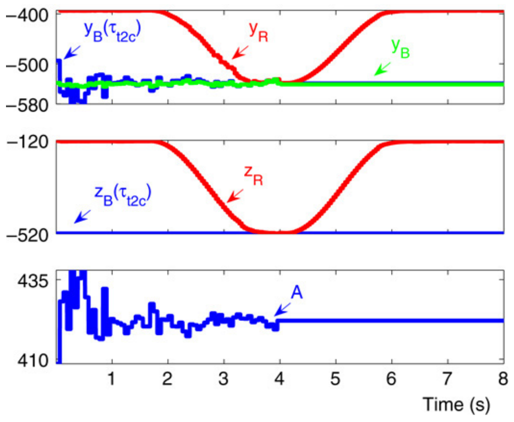


Fig. 11. y_R and z_R illustrate the timed trajectory as recorded by the Puma arm for the situation depicted in Fig. 9. Initially, the y coordinate for point-to-contact, $y_B(\tau_{t2c})$, and the y_B coordinate of ball trajectory, as acquired by the visual system are quite noisy. The robot trajectories and the calculated coordinates for contact coincide after movement time, and the ball is successfully caught.

The hysteresis property allows for a special kind of behavioral stability that leads to a simple kind of memory which determines system performance depending on its past history and enables the system to be robust to ambiguity in the environment.

4.2.2. Sensory conditions for ball interception become invalid

A globally optimized behavior results from the coupling of the different behavioral modules which define the overall dynamics (Eq. (1)). By obeying the time scale separation principle for specification of speed of convergence of the different dynamical systems, the system is designed such that the final behavior results from local sensor control and global task constraints without a representation of the entire behaviors the system can perform. This decision making mechanism allows to intelligently combine information from multiple sources and select appropriate behavior according to the environment situation.

Specifically, the system is expected to catch a ball exhibiting the following sequence of behaviors or events: (1) stopped at the arm resting position while waiting for movement initiation; (2) moving from this position towards the ball catching position; and (3) moving back to arm resting position after ball catching. However, sequence generation of behaviors depends on the local sensory information and on the system state. This responsiveness and flexibility is the major benefit of the proposed approach compared to conventional techniques.

The system is able to make decisions such that it flexibly responds to the demands of any given situation while keeping timing stable. The decision is dependent on local information available at the system's current position. This is achieved by obeying the principles of the Dynamic Approach and illustrates the power of our approach: the behavior of the system itself leads to the changing sensor information which controls the change and persistence of a rich set of behaviors.

The design of the quasi-boolean parameters of the competitive dynamics guarantees that flexibility is fulfilled: if the sensorial context changes such that the previously generated sequence is no longer adequate, the plan is changed and a new sequence of events emerges.

Thus, when sensory conditions change an appropriate new sequence of events emerges. When one of the sensory conditions for ball interception is invalid (e.g., ball becomes invisible, unreachable, or no longer approaches with appropriate time-to-contact), then one of the following happens depending on the point within the sequence of events at which the change occurs: (1) If the change occurs during the initial postural stage, the system stays in

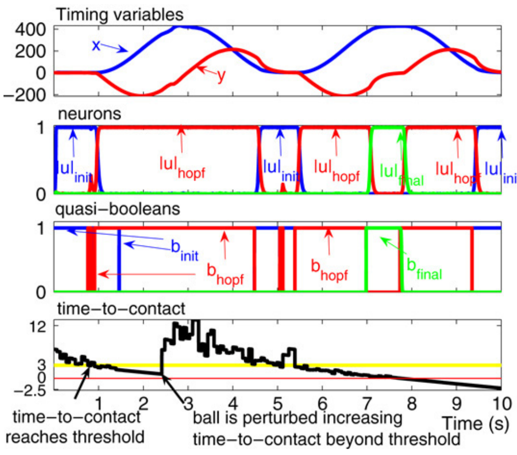


Fig. 12. Similar to Fig. 11, but the ball is suddenly shifted at about 2.4 time units leading to a time-to-contact larger than the threshold value (3 s) required for movement initiation. The arm is in the motion stage at this point and thus continues its movement a full cycle, until captured by the initial postural state when the arm is back to the reference position.

that postural state. (2) If the change occurs during the movement, then the system continues on its trajectory, now going around a full cycle to return to the reference posture. (3) When the change occurs during posture in the target position, a discrete movement is initiated that takes the arm back to its resting position.

Perturbations at the time-to-contact

Consider the ball is suddenly shifted away from the arm at about 2.4 time units, leading to much larger time-to-contact, well beyond threshold for movement initiation. The arm is in the motion stage at this point and hence continues its movement a full cycle, until captured by the initial postural state when the arm is back to the reference position. At the perturbation time, u_{hopf} neuron is activated (Fig. 12) and the other neurons are deactivated. The u_{hopf} neuron rests activate while the arm continues its movement a full cycle. At the time the m dynamical variable is captured by the initial postural state ($m = 0$), the quasi-boolean b_{init} becomes one, triggering the activation of the neuron, u_{init}, and b_{hopf} becomes zero, deactivating the corresponding neuron u_{hopf}. The arm rests in the reference position. This behavior emerges from the sensory conditions controlling the neuronal dynamics. However, because sensory conditions are appropriate, a new movement is initiated and the ball is successfully caught.

Fig. 13 shows that a new sequence of events emerges when the change occurs during posture in the final posture position (at approximately 3.5 time units). This change invalidates the sensorial context for quasi-boolean b_{final} which changes to zero while b_{hopf} goes to one. The neurons switch accordingly and a discrete movement is initiated backwards taking the arm back to its resting position.

Perturbations at the point-to-contact

If the ball becomes unreachable another type of sensorial condition change occurs. At about 1.9 time units, the ball is suddenly shifted away from the arm leading to a point-to-contact no longer reachable. Fig. 14 shows how the arm rests in the reference position when the change occurs during motion stage but still in the vicinity of initial posture state.

When reachability condition changes during the periodic motion phase, the system continues its periodic motion, as shown in Fig. 15. These experiments demonstrate that sequence generation is stably adapted to changing online sensory information.

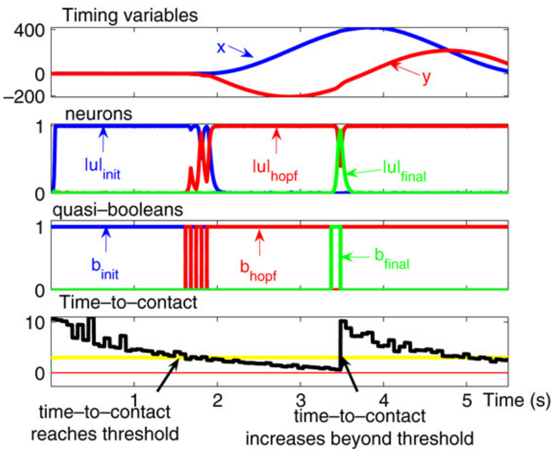


Fig. 13. Similar to Fig. 11, but time-to-contact change occurs at about 3.5 time units. A discrete movement is initiated taking the arm back to its resting position.

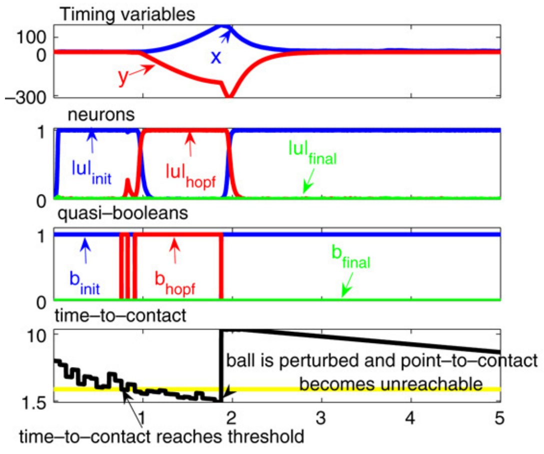


Fig. 14. The ball is suddenly shifted away from the arm at about 1.9 time units, leading to an unreachable position, out of the robot workspace. The arm is still in the vicinity of the initial posture state and rests in the reference position.

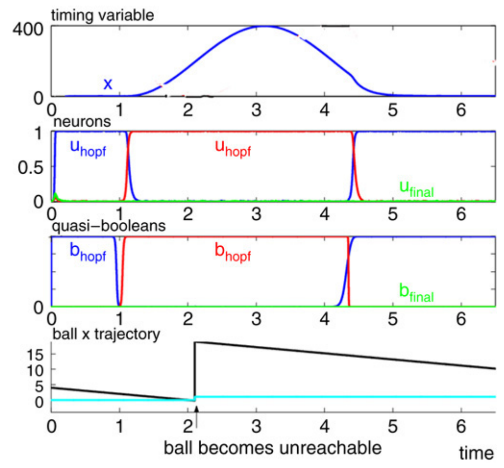


Fig. 15. Similar to Fig. 14, but reachability condition change occurs at about 2 time units, leading to an unreachable point-of-contact. The arm continues in the motion stage at this point.

5. Conclusion/Outlook

In this article, an attractor based dynamics autonomously generated temporally discrete movements and movement sequences for a Puma arm. The task was to generate a timed movement from an initial posture to catch an approaching ball moving in a complex, unstructured environment. After catching the ball or in case catching becomes impossible, the arm moves back to its resting position, ready to initiate a new movement whenever appropriate sensory information arrives. Movement initiation and termination was entirely sensor driven and autonomous sequence generation was stably adapted to changing unreliable online visual sensory information. Ball tracking was robustly achieved by applying a CAMSHIFT algorithm [5] to the visual sensory information acquired by an inexpensive FireWire camera.

The implementation described provides a rigorous test of the dynamic architecture robustness and probes how its inherent stability properties play out when the sensory information is noisy and unreliable.

Results enable us to positively answer the two questions asked in the introduction. The first was if timed trajectories comprising sequence generation can be flexibly generated. Results illustrate the dynamic architecture robustness and show that synchronization properties of this type of oscillator can be exploited to successfully generate sequence and timed actions. This flexibility property was illustrated in real experiments. Some basic inherent properties of structurally stable dynamical systems, such as stability, bifurcation, and hysteresis provide the ability to modify online the generated attractor landscape to the demands of the current situation, depending on the sensorial context. In other words, we have explored the fact that the resultant behavior is a combination of the intrinsic dynamics and external input. Stability and controllability of the overall system was guaranteed by obeying the time scale separation principle. Other distinguishable feature of the proposed approach, is the analytically solvability, which facilitates the specification of parameters. This was exploited providing a theoretically based way of tuning the movement parameters, such as movement time, movement extent, etc, such that it is possible to account for relationships among these. Smooth trajectory online modulation of the trajectories with respect to the goal, amplitude and frequency is now possible according to the environmental changes, such that action is steered by online modulation of the parameters. Thus, there is an independence relative to the specification of individual movement parameters. Also, a globally optimized behavior was achieved through local sensor control and global task constraints.

The second question asked was if these generated trajectories can be stably and robustly implemented in a robot arm with modest computational resources. Results show that this approach does not place unreasonable constraints on the environment in which the robot operates and assures a quick reaction to eventual changes in the sensed environment.

The approach shows up several appealing properties, such as perception-action coupling and reusability of the primitives. In the field of robotics, the proposed approach holds the potential to become a much more powerful strategy for generating complex movement behavior for systems with several degrees-of-freedom (DOFs) than classical approaches. This type of control scheme has a great potential for generating robust locomotion and movement controllers for robots.

Currently, we are addressing the approach extension to robust locomotion generation and movement controllers for robots as this framework finds a great number of applications in service tasks and seems ideal to achieve intelligent and more human like prostheses.

References

[1] P. Fua, A. Shahrokni, T. Drummond, Texture boundary detection for real-time tracking, in: Proc. Eur. Conf. on Computer Vision, vol. 2, Prague, Czech Republic, May 2004, pp. 566–577.

[2] M. Shah, A. Yilmaz, L. Xin, Contour-based object tracking with occlusion handling in video acquired using mobile cameras, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (11) (2004) 1531–1536.

[3] M.A. Arbib, Perceptual structures and distributed motor control, in: V.B. Brooks (Ed.), Handbook of Physiology. Sect. 1: The Nervous System. Vol. II Motor Control. Part 2, American Physiological Society, Bethesda, Maryland, 1981, pp. 1449–1480.

[4] R. Blickhan, The spring-mass model for running and hopping, Journal of Biomechanics 22 (11–12) (1989) 1217–1227.

[5] G.R. Bradski, Computer vision face tracking as a component of a perceptual user interface, in: Workshop on Applications of Computer Vision, Princeton, NJ, October 1998, pp. 214–219.

[6] R.A. Brooks, New approches to robotics, Science 253 (1991) 1227–1232.

[7] M. Bühler, D.E. Koditscheck, R.D. Skinner, Planning and control of a juggling robot, International Journal of Robotics Research 13 (2) (1994) 101–118.

[8] M. Clark, G. Anderson, R. Skinner, Coupled oscillator control of autonomous mobile robots, Autonomous Robots 9 (2000) 189–198.

[9] F. Delcomyn, Neural basis for rhythmic behaviour in animals, Science 210 (1980) 492–498.

[10] M. Erdmann, T. Lozano-Perez, On multiple moving objects, Algorithmica 2 (1987) 477–521.

[11] Paolo Fiorini, Zvi Shiller, Time optimal trajectory planning in dynamic environments, in: Proceedings of the IEEE International Conference on Robotics and Automation, vol. 2, Minneapolis, MN, US, April 1996, pp. 1553–1558.

[12] Th. Fraichard, Trajectory planning in a dynamic workspace: A "state-time space" approach, Advanced Robotics 13 (1) (1999) 75–94.

[13] K.S. Fu, R.C. Gonzalez, C.S.G. Lee, Robotics: Control, Sensing, Vision, and Intelligence, McGraw-Hill, New York, 1987.

[14] K. Fujimura, H. Samet, A hierarchical strategy for path planning among moving obstacles, IEEE Transaction on Robotics and Automation 5 (1) (1989) 61–69.

[15] Y. Fukuoka, H. Kimura, A.H. Cohen, Adaptive dynamic walking of a quadruped robot on irregular terrain based on biological concepts, The International Journal of Robotics Research 3–4 (2003) 187–202.

[16] S.F. Giszter, F.A. Mussa-Ivaldi, E. Bizzi, Convergent force fields organized in the frog's spinal cord, Journal of Neuroscience 13 (1993) 467–491.

[17] R. Glasius, A. Komoda, S. Gielen, Population coding in a neural net for trajectory formation, Network: Computation in Neural Systems 5 (1994) 549–563.

[18] A.J. Ijspeert, J. Nakanishi, S. Schaal, Learning control policies for movement imitation and movement recognition, in: Neural Information Processing System, NIPS2001, 2001.

[19] S. Kotosaka, S. Schaal, Synchronized robot drumming by neural oscillator, in: The International Symposium on Adaptive Motion of Animals and Machines, Montreal, Canada, 2000.

[20] E. Large, H. Christensen, R. Bajcsy, Scaling the dynamic approach to path planning and control: Competition amoung behavioral constrains, International Journal of Robotics Research 18 (1) (1999) 37–58.

[21] D. Koditschek, M. Bühler, Kindlmann, Planning and control of a juggling robot, International Journal of Robotics Research 13 (2) (1994) 101–118.

[22] B. Thomas, M. Everingham, Supervised segmentation and tracking of non-rigid objects using a mixture of histograms, in: Proceedings of the 8th IEEE International Conference on Image Processing, Thessaloniki, Greece, October 2001, pp. 62–65.

[23] E. Marchand, M. Pressigout, Real-time planar structure tracking for visual servoing: A contour and texture approach. in: Int. Conf. on Intelligent Robots and Systems, vol. 2, Edmonton, Canada, August 2005, pp. 1701–1706.

[24] R. Pfeiffer, Building "fungus eaters": Design principles of autonomous agents, in: P. Maes, M. Mataric, J.-A. Meyer, J. Pollack, S.W. Wilson (Eds.), From Animals to Animats: Proc. of the 4 Int. Conf. on Simulation and Adaptive Behavior, MIT Press, Cambridge, MA, 1996, pp. 3–12. A Bradford Book.

[25] H. Tao, Q. Zhao, Object tracking using color correlogram, in: IEEE Workshop on VS-PETS, Beijing, China, October 2005, pp. 263–270.

[26] M.H. Raibert, Legged Robots That Balance, MIT Press, Cambridge, MA, 1986.

[27] J. Reif, M. Sharir, Motion planning in the presence of moving obstacles. in: Proceedings of the 25th IEEE Symposium on the Foundation of Computer Science, Portland, OR, USA, October 1985, pp. 144–153.

[28] C. Santos, Generating timed trajectories for an autonomous vehicle: A non-linear dynamical systems approach, in: IEEE Int. Conf. on Robotics and Automation, IEEE, New Orleans, 26April–1 May 2004, pp. 3741–3746.

[29] C.P. Santos, M. Ferreira, Two vision-guided vehicles: Temporal coordination using nonlinear dynamical systems, in: Proceedings of the 2007 IEEE International Conference on Robotics and Automation, ICRA 2007, Roma, Italy, 10–14 April 2007, pp. 14–19.

[30] Cristina Santos, Jaime Fonseca, Paulo Garrido, Carlos Couto, Surface profile based on sensor fusion, in: Proceedings of the 6th UK Mechatronics, Skovde, Switzerland, 1999, pp. 613–619.

[31] Cristina P. Santos, Cutting edge robotics, in: Navigation Section. Generating Timed Trajectories for Autonomous Robotic Platforms. A Non-Linear Dynamical Systems Approach, 2005, pp. 255–278 (Chapter III).

[32] S. Schaal, S. Kotosaka, D. Sternad, Nonlinear dynamical systems as movement primitives, in: International Conference on Humanoid Robotics, 6–7 Sept, 2001, Springer, Cambridge, MA, 2001, pp. 117–124.

[33] G. Schöner, Dynamic theory of action- perception patterns: The time-before-contact-paradigm, Human Movement Science 3 (1994) 415–439.
[34] G. Schöner, M. Dose, A dynamical systems approach to task-level system integration used to plan and control autonomous vehicle motion, Robotics and Autonomous Systems 10 (1992) 253–267.
[35] G. Schöner, C. Santos, Control of movement time and sequential action through attractor dynamics: A simulation study demonstrating object interception and coordination, in: 9th Intelligent Symp. on Intelligent Robotic Systems, Toulouse, France, 18–20, July 2001.
[36] A. Steinhage, G. Schöner, Dynamical systems for the behavioral organization of autonomous robot navigation, in: Spie-Intelligent Sys. Manufactors, 1998, pp. 169–180.
[37] G. Taga, Emergence of bipedal locomotion through entrainment among the neuro-musculo-skeletal system and the environment, Physica D: Nonlinear Phenomena 75 (1–3) (1994) 190–208.
[38] J. Tani, Y. Ito, Y. Sugita, Self-organization of distributedly represented multiple behavior schemata in a mirror system: Reviews of robot experiments using rnnpb, Neural Networks 17 (2004) 1273–1289.
[39] H. Tuan, T. Son, P. Apkarian, T. Nguyen, Low-order iir filter bank design, Circuits and Systems I: IEEE Transactions on 8 (1) (2005) 1673–1683.
[40] M. Williamson, Rhythmic robot arm control using oscillators, in: IEEE Int. Conf. on Int. Robots and Systems, October 1998.
[41] S. Grillner, Modular organization of spinal motor systems, Neuroscientist 8 (5) (2002) 437–442.
[42] E. Bizzi, A. DAvella, P. Saltiel, M. Tresch, Neural networks for vertebrate locomotion, Scientific American (1996) 48–53.

Cristina Santos received the B.S. degree in Industrial Electronics, the M.Sc. degree in Robotics, and the Ph.D. degree in Robotics in the field of Nonlinear dynamics, all from University of Minho, Guimaraes, Portugal, in 1994, 1998 and 2003 respectively. The Ph.D. was also in collaboration wht the CNRS-CNRC Marseille, France. She is currently working as an Auxiliar Professor at the University of Minho, Industrial Electronics Department. The main interests lie on the nonlinear dynamical systems approach to the behavior generation in the robotics domain. Her research focus on the extension of the use of the dynamical systems theory to the achievement of more complex behavior for robots: generate locomotion for multi-dof robots and to achieve cooperativity among multi-robots.

Manuel Ferreira received the B.S. degree in Electronics and Telecommunications from University of Aveiro, in 1992. He received the M.Sc. degrees in Industrial Informatics and the Ph.D. degree in Industrial Electronics from University of Minho in 1996 and 2004, respectively. He is currently working as Professor at the University of Minho, Industrial Electronics Department mainly teaching in the fields of image and signal processing. The main research interests lie on computer vision, image processing and image analysis. He has been working mainly in the development of computer vision technologies based on advanced algorithms, applied to industrial applications, motion analysis and human–robot interface.