Ball Catching by a Puma Arm: a Nonlinear Dynamical Systems Approach

Cristina Santos
Industrial Electronics Department
University of Minho
Guimaraes, Portugal
Email: cristina@dei.uminho.pt

Manuel Ferreira
Industrial Electronics Department
University of Minho
Guimaraes, Portugal
Email: mjf@dei.uminho.pt

***Abstract*— We present an attractor based dynamics that autonomously generates temporally discrete movements and movement sequences 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 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. We demonstrate that the implemented decision-mechanism is robust to noisy sensorial information. Further, 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.

I. Introduction

This article addresses the problem of generating timed trajectories and sequences of movements for robotic manipulators when relatively low-level, noisy sensorial information is used to initiate and steer action. The developed architectures are fully formulated in terms of nonlinear dynamical systems which lead to a flexible timed behaviour stably adapted to changing online sensory information. The generated trajectories have controlled and stable timing (limit cycle type solutions). Incoupling of sensory information enables sensor driven initiation and termination of movement.

Specifically, we address the following question: Is it possible to flexibly generate timed trajectories comprising sequence generation and stably and robust implement them in a robot arm with modest computational resources? Flexibility means here that if the sensorial context changes such that the previously generated sequence is no longer appropriated a new sequence of behaviours, adequate to the current situation, emerges.

This question is positively answered and shown in an exemplary situation in which a PUMA robot arm catches a moving object and returns to a reference position thereafter. The evaluation results illustrate the stability and flexibility properties of the dynamical architecture as well as the robustness of the decision-making mechanism implemented.

We build on previous work [1]–[4], where we have shown that the proposed approach is sufficient versatile to generate, through limit cycle attractors, a whole variety of richer forms of behavior, including both rhythmic and discrete tasks. The online linkage of the generated timed trajectories to online noisy sensorial information, was achieved through the coupling of several nonlinear dynamical systems [1], [4]. In [1], this architecture was implemented in a real vehicle. Further, we have shown that the proposed approach can be integrated with other dynamical architectures which do not explicitly parameterize timing requirements [1], [3].

This work is innovative in the manner how it formalizes and uses movement primitives, both in the context of biological and robotics research. Further, it significantly facilitates generation of movement and sequences of movements. We apply autonomous differential equations to model the manner how behaviors related to locomotion are programmed in the oscillatory feedback systems of ”central pattern generators” in the nervous systems [4].

The idea of using dynamic systems for movement generation is not new. For instance, the *Dynamical Systems approach to autonomous robotics*, has been developed for the control of autonomous vehicles [5], [6]. Other solutions [7]–[11] 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 [7] 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, [12] has been able to generate temporally discrete movement as well. The flexible activation of different motor acts in response to user demands or sensed environmental conditions is more difficult to achieve from the control level.

The work presented in this article extends the use of

oscillators to tasks on an arm robot. It also differs from most of the literature in that it is implemented on a real robot.

II. Timed trajectories generation

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 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

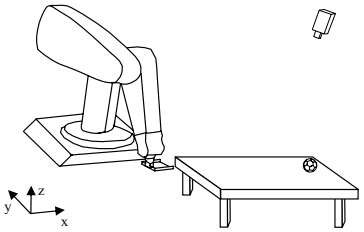

Fig. 1. The Puma arm must catch the ball before it falls over the table. The x and z coordinates for catching position are fixed and known. A camera acquires visual information that enables the system to calculate the y coordinate of catching position and the time it takes for the ball to be at that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

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.

The three relevant coordinate systems involved in this task formulation must be linked: (a) The dynamical variable coordinate system, in which temporal movement is planned, describes the x dynamical variable position along a straight path from an initial to a final posture position. (b) The task reference coordinate system describes the end-effector position, (x, y, z), of the arm along a straight path from the initial end-effector position (initial posture) to the ball catching position. (c) The arm kinematics is described by six joint angles [13].

Frame (a) and (b) are linked through straightforward formulae, which depend on the calculated ball catching position (point-to-contact). Frame (b) and (c) are linked through the kinematic model of the robot arm and its inverse (based on the geometrical solution [13]). A starting end-effector orientation is established and kept constant during motion. During movement execution, the dynamical variables are continuously transformed into task frame (b), from which joint angles are computed through the inverse kinematic transformation.

A. The dynamical systems trajectory generator

In order to formulate this task using the nonlinear dynamical systems approach, we model rhythmic and discrete movements as a time course of behavioral variables (x, y) generated by dynamical systems. Trajectories are generated as stable solutions of the following dynamical system, which generates both stable oscillations and two stationary states [2],

$$\begin{aligned} \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} &= 5\ |u_{\rm init}| \begin{pmatrix} x \\ y \end{pmatrix} + |u_{\rm hopf}| f_{\rm hopf} \\ &+ 5\ |u_{\rm final}| \begin{pmatrix} x - A \\ y \end{pmatrix} + {\rm gwn}. \end{aligned} \tag{1}$$

Although only x will be used to control motion of a relevant robotic task variable, a second auxiliary variable, y, is needed to enable the system to undergo periodic motion.

A neural dynamics controls the switching between the three regimes through three "neurons" $u_{\rm i}$ ($i = {\rm init}, {\rm hopf}, {\rm final}$).

Herein, an approach is defined to cope with fluctuations in amplitude of calculated point-to-contact by including in the vector field quantities that depend on sensory information. This is achieved in two steps. Firstly, the "init" and "final" contributions generate stable stationary solutions at $x = 0$ for "init" and A for "final" with $y = 0$ for both. These states are characterized by a time scale of $\tau = 1/5 = 0.2$.

Secondly, the "Hopf" contribution (as defined in [2]) is modified as follows:

$$\mathbf{f}_{\rm hopf} = \begin{pmatrix} \alpha & -\omega \\ \omega & \alpha \end{pmatrix} \begin{pmatrix} \left(x - \frac{A}{2}\right) \\ y \end{pmatrix} \tag{2}$$

$$- \gamma \left(\left(x - \frac{A}{2}\right)^2 + y^2 \right) \begin{pmatrix} \left(x - \frac{A}{2}\right) \\ y \end{pmatrix} \tag{3}$$

where $\gamma = \frac{4\ \alpha}{A^2}$ defines amplitude of Hopf contribution. This "Hopf" contribution, as described in [1], [2], models the oscillatory movement between two x values and contains a bifurcation from a fixed point to a limit cycle. We use it because it can be completely solved analytically, providing complete control over its stable states. 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 dynamics are augmented by a Gaussian white noise term, gwn, that guarantees escape from unstable states and assures robustness to the system.

1) Neural dynamics: The "neuronal" dynamics of $u_{\rm i} \in [-1, 1]$ ($i = {\rm init}, {\rm final}, {\rm hopf}$) switches the dynamics from the initial and final posture states into the oscillatory regime and back. The competitive dynamics are given by

$$\alpha_u\, \dot{u}_{\rm i} = \mu_{\rm i}\ u_{\rm i} - |\mu_{\rm i}|\ u_{\rm i}^3 - 2.1 \sum_{a \neq i} u_{\rm a}^2\ u_{\rm i} + {\rm gwn}, \tag{4}$$

where "neurons" can go "on" (=1) or "off" (=0). 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) [14].

In order to control switching, these parameters are explicitly designed such that their functions reflect the current sensorial context and the global constraints expressing which states are more applicable to the current situation. They are defined as functions of user commands, sensory events, or internal states and control the sequential activation of the different neurons (see [6], for a general framework for sequence generation based on these ideas and [2] 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 $\tau_u = 0.02$, ten times faster than the relaxation time of the x, y dynamical variables. The adiabatic elimination of fast behavioral variables reduces the complexity of a complicated behavioral system built up by coupling many dynamical systems [3], [6]. By using different time scales one can design the several dynamical systems separately.

B. Coupling to sensorial information

In order to intercept an approaching ball it is necessary to be at the right location at the right time. This information is acquired by a camera mounted in the ceiling over the robot workspace.

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 [15]–[17].

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.

Although conventional single-cue 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 [18], that handles the described computer-vision application problems during its operation.

The CAMSHIFT algorithm tracks the x', y' image coordinates and area of the color blob representing the green ball. A perspective projection model transforms the x', y' image coordinates onto the x, y world coordinates.

In this application, the robot arm catches the ball at the end of the table in which the ball is moving (see Fig. 1). The time it takes to the ball to intersect the arm at this point in space, that is, the time-to-contact, t_{2c}, is extracted from the obtained visual information through straightforward formulae, by considering the ball has a linear trajectory in the 3D cartesian space with a constant approach velocity. The point-to-contact, $(x(\tau_{\mathrm{t2c}}), y(\tau_{\mathrm{t2c}}), z(\tau_{\mathrm{t2c}}))$, is computed along similar lines.

For simplicity, $x(\tau_{\mathrm{t2c}})$ is always constant and has a known value ($x(\tau_{\mathrm{t2c}}) = -154$ mm). Further, $z(\tau_{\mathrm{t2c}})$ corresponds to the table's height and thus $z(\tau_{\mathrm{t2c}})) = -519$ mm.

Movement amplitude, A, is set as,

$$A = \sqrt{(y(\tau_{\mathrm{t2c}}) - y_{\mathrm{R}}(0))^2 + (z(\tau_{\mathrm{t2c}}) - z_{\mathrm{R}}(0))^2}, \quad (5)$$

where $y_{\mathrm{R}}(0)$ and $z_{\mathrm{R}}(0)$ denote end-effector position previously to movement initiation.

C. 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 ($b_{\mathrm{x_R\ close\ x_{init}}}(x)$); **and** either of the following is true: (1) ball not approaching or not visible ($\tau_{\mathrm{t2c}} \leq 0$); (2) ball contact not yet within a criterion time-to-contact ($\tau_{\mathrm{t2c}} > \tau_{\mathrm{crit}}$); (3) ball is approaching within criterion time-to-contact but is not reachable ($0 < \tau_{\mathrm{t2c}} < \tau_{\mathrm{crit}}$; $b_{\mathrm{reachable}} = 0$); (4) ball stopped ($b_{\mathrm{stopped}} = 1$); (5) ball was catched ($b_{\mathrm{catched}} = 1$); (6) ball has disappeared ($b_{\mathrm{disappeared}} = 1$).

The factor $b_{\mathrm{x_R\ close\ x_{init}}}(x_{\mathrm{robot}}) = \sigma(0.15\ \mathrm{A} - x_{\mathrm{robot}})$ 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

$$\sigma(x) = [\tanh(100x) + 1]/2, \quad (6)$$

although any other functional form will work as well.

These logical conditions can be expressed through this mathematical function:

$$\begin{aligned} b_{\mathrm{init}} &= \sigma(0.15A - x_{\mathrm{R}})\,[\sigma(\tau_{\mathrm{t2c}} - \tau_{\mathrm{crit}}) + \sigma(-\tau_{\mathrm{t2c}}) \\ &+ \sigma(\tau_{\mathrm{t2c}})\ \sigma(\tau_{\mathrm{crit}} - \tau_{\mathrm{t2c}})\ \sigma(1 - b_{\mathrm{reachable}}) \\ &+ b_{\mathrm{stopped}} + b_{\mathrm{catched}} + b_{\mathrm{disappeared}}]\,. \end{aligned} \quad (7)$$

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

A similar analysis derives the b_{hopf} and b_{final} parameters, but the "or" is expressed with the help of the "not" (subtracting from 1) and the "and":

$$\begin{aligned} b_{\mathrm{hopf}} \;=\; & 1-(1-[\sigma(0.15A-x_{\mathrm{R}})\;\sigma(\tau_{\mathrm{t2c}})\;\sigma(\tau_{\mathrm{crit}}-\tau_{\mathrm{t2c}}) \\ & \sigma(b_{\mathrm{reachable}})])\cdot(1-[\sigma(x_{\mathrm{R}}+0.85A) \\ & \{\sigma(1-b_{\mathrm{reachable}})+\sigma(-\tau_{\mathrm{t2c}})+\sigma(\tau_{\mathrm{t2c}}-\tau_{\mathrm{crit}}) \\ & +\sigma(0.85A-x_{\mathrm{R}})+\sigma(b_{\mathrm{stopped}})+\sigma(b_{\mathrm{catched}}) \\ & +\;\sigma(b_{\mathrm{disappeared}})\}]) \end{aligned} \tag{8}$$

$$\begin{aligned} b_{\mathrm{final}} \;=\; & \sigma(\tau_{\mathrm{t2c}})\;\sigma(\tau_{\mathrm{crit}}-\tau_{\mathrm{t2c}})\;\sigma(b_{\mathrm{reachable}}) \\ & \sigma(x_{\mathrm{R}}-0.85A)\;\sigma(1-b_{\mathrm{stopped}})\;\sigma(1-b_{\mathrm{disappeared}}) \end{aligned} \tag{9}$$

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

The puma velocity, V_{puma}, as planned by the dynamical system is set as:

$$V_{\mathrm{puma}} = |\dot{x}|, \tag{10}$$

where x is the dynamical variable.

Previously to timed trajectory generation, the robot arm is moved to a pre-defined location, $(x_{\mathrm{R}}, y_{\mathrm{R}}, z_{\mathrm{R}})(0) = (-154, -393, -123)$ mm, whose x coordinate equals $x(\tau_{\mathrm{t2c}})$. Therefore, robot arm movement only happens in the Z and Y plane. The x dynamical variable is then converted by simple coordinate transformations onto the y_{R} and z_{R} coordinates of robot movement.

III. Experimental results

The dynamic architecture was implemented and evaluated on a PUMA arm. 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) [19] and a personal computer (PC). 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.

The CCD color cameras are FireWire with a resolution of 640x480 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. 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 2ms (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 2ms 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 2s.

A. Properties of the generated timed trajectory

Fig. 2 shows the time courses of the relevant variables and parameters when the PUMA arm successfully catches an approaching ball. As the ball approaches, the current time-to-contact becomes smaller than a critical value (here 3s), 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 (x 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 real point-to-contact is at $(x, y, z) = (-154, -553, -519)$ mm which yields a movement amplitude of 427mm. Noisy sensory information produces amplitude fluctuations in point-to-contact. However, these fluctuations are included in the vector field and thus are filtered.

The arm waits in the final posture. Ball catching is detected by the vision system which activates the b_{catched} flag and leads to autonomous initiation of the backward movement to the arm resting position.

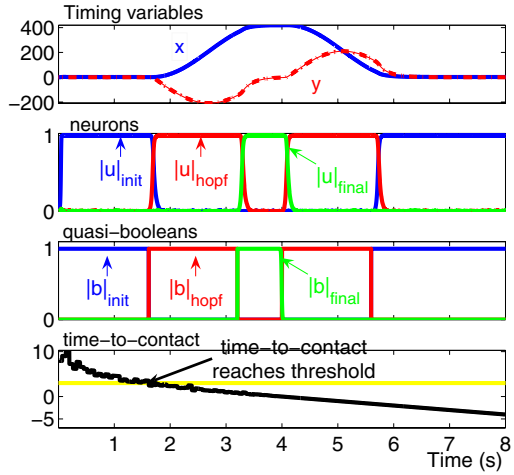

Fig. 2. Trajectories of variables and parameters in autonomous ball catching and return to resting position. The top three panels represent timing variables, neural variables and quasi-booleans. The bottom panel shows the time-to-contact, which crosses a threshold at about 3 time units. At this moment, the arm initiates its timed movement.

The y_R and z_R real robot trajectories are illustrated in Fig. 3. Despite the noisy amplitude, the robot trajectories are almost not affected.

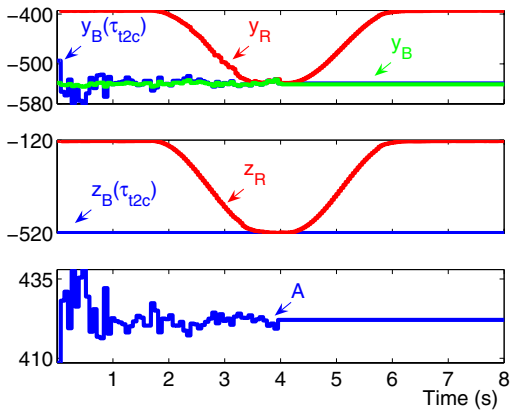

Fig. 3. y_R and z_R illustrate the timed trajectory as recorded by the Puma arm for the situation depicted in Fig. 2. Initially, the y coordinate for point-to-contact, $y_B(\tau_{t2c})$, and the y_B coordinate of ball trajectory, as acquired by the visual system are quite noisy. The robot trajectories and the calculated coordinates for contact coincide after movement time, and the ball is successfully catched.

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 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 [2]), the mean value of $d_{\text{collision}}$ was 100 mm. The proposed solution leads to improvement.

The fact that timed movement is generated from attractor solutions of a nonlinear dynamical system leads to a number of properties of this system, that are potentially useful to other real-world implementations of this form of autonomy. The experiment shown in Fig. 2, 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. Note how the autonomous sensor-driven initiation of movement is stabilized by the 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 [2], [5].

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.

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 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.

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.

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 (Fig. 4). 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. 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 catched.

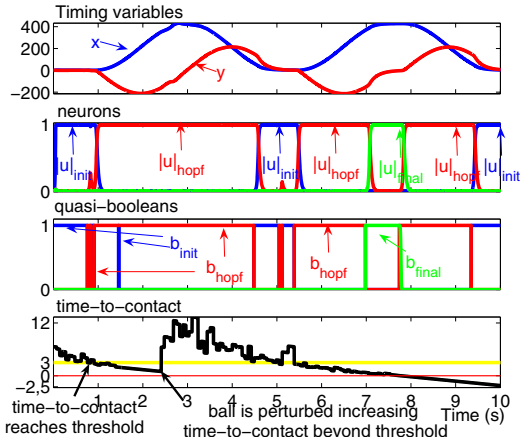

Fig. 4. Similar to Figure 4, but the ball is suddenly shifted at about 2.4 time units leading to a time to contact larger than the threshold value (3s) required for movement initiation.

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. 5 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.

IV. 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

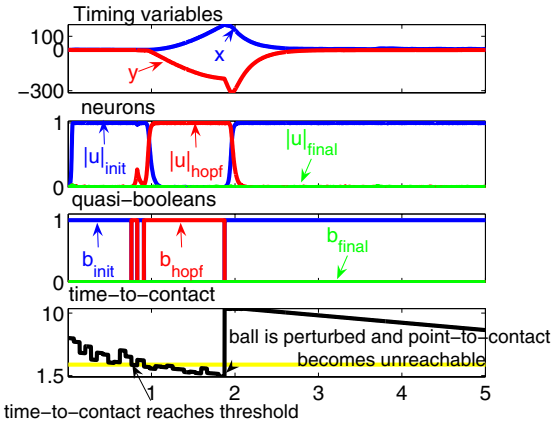

Fig. 5. The ball is suddenly shifted away from the arm at about 1.9 time units, leading to an unreachable position, out of the robot workspace. The arm is still in the vicinity of the initial posture state and rests in the reference position.

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 [18] to the visual sensory information acquired by an unexpensive FireWire camera.

The described implementation 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.

The attractor based dynamics was based on previous work [2], but the dynamic architecture was changed in order to cope with unreliable sensor information by including this information in the vector field.

The dynamical approach has various desirable properties. Firstly, its inherent properties, 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. This flexibility property was illustrated in real experiments. Secondly, a globally optimized behavior is achieved through local sensor control and global task constraints. Further, we guarantee the stability and the controllability of the overall system by obeying the time scale separation principle. Finally, 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 analytically solvability and the generalization to sequence generation, are two distinguishable features of our approach.

Future work will address how to extend the described model to achieve more complex behavior for systems with several degrees-of-freedom. We will address 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] C. Santos, "Generating timed trajectories for an autonomous vehicle: A non-linear dynamical systems approach," in *IEEE International Conference on Robotics and Automation (ICRA)*. New Orleans, LA U: IEEE, April 26-1 May 2004, pp. 3741–3746.

[2] G. Schöner and C. Santos, "Control of movement time and sequential action through attractor dynamics: A simulation study demonstrating object interception and coordination," in *9th Intelligent Symposium on Intelligent Robotic Systems - SIRS'2001*, Toulouse, France, 18-20,July 2001.

[3] C. P. Santos, *Cutting Edge Robotics*, 2005, ch. III. Navigation Section. Generating Timed Trajectories for Autonomous Robotic Platforms. A Non-Linear Dynamical Systems Approach, pp. 255–278.

[4] G. Schöner, "Dynamic theory of action - perception patterns: The time-before-contact-paradigm," *Human Movement Science*, vol. 3, pp. 415–439, 1994.

[5] G. Schöner and M. Dose, "A dynamical systems approach to task-level system integration used to plan and control autonomous vehicle motion," *Robotics and Autonomous Systems*, vol. 10, pp. 253–267, 1992.

[6] A. Steinhage and G. Schöner, "Dynamical systems for the behavioral organization of autonomous robot navigation," in *Sensor Fusion and Decentralized Control in Robotic Systems: Proceedings of Spie-Intelligent Systems Manufactors*, I. M. G. T. S. PS, Ed. Boston: SPIE-publishing, 1998, pp. 169–180.

[7] M. H. Raibert, *Legged robots that balance*. Cambridge, Massachusetts: MIT Press, 1986.

[8] S. Schaal and C. G. Atkeson, "Open loop stable control strategies for robot juggling," in *IEEE International Conference on Robotics and Automation*, Georgia, Atlanta, 1990, pp. 913–918.

[9] D. E. K. M Bühler and Kindlmann, "Planning and control of a juggling robot," *International Journal of Robotics Research*, vol. 13, no. 2, pp. 101–118, 1994.

[10] M. R. Clark, G. T. Anderson, and R. D. Skinner, "Coupled oscillator control of autonomous mobile robots," *Autonomous Robots*, vol. 9, pp. 189–198, 2000.

[11] M. Williamson, "Rhythmic robot arm control using oscillators," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'98)*, Victoria, B.C., Canada, October 1998.

[12] S. Schaal, S. Kotosaka, and D. Sternad, "Nonlinear dynamical systems as movement primitives," in *IEEE International Conference on Humanoid Robotics*. IEEE, Cambridge, MA, 2000.

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

[14] E. W. Large, H. I. Christensen, and R. Bajcsy, "Scaling the dynamic approach to path planning and control: Competition amoung behavioral constrains," *International Journal of Robotics Research*, vol. 18, no. 1, pp. 37–58, 1999.

[15] E. M. M. Pressigout, "Real-time planar structure tracking for visual servoing: A contour and texture approach." Int. Conf. on Intelligent Robots and Systems, IROS'05, August 2005.

[16] L. K. G. Taylor, "Fusion of multimodal visual cues for model-based object tracking." Brisbane: Australasian Conference on Robotics and Automation, December 2003, pp. 1–3.

[17] B. T. M. Everingham, "Supervised segmentation and tracking of non-rigid objects using a "mixture of histograms." Proceedings of the 8th IEEE International Conference on Image Processing (ICIP 2001), October 2001, pp. 62–65.

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

[19] C. Santos, J. Fonseca, P. Garrido, and C. Couto, "Surface profile based on sensor fusion," in *Proceedings of the 6th UK Mechatronics*, Skovde, Switzerland, 1999, pp. 613–619.