Organizing and flexibly updating timed actions: An attractor dynamics approach

Farid Oubbati, Mathis Richter, Gregor Schöner
Institut für Neuroinformatik, Ruhr-Universität Bochum
Universitätsstr. 150, 44780 Bochum, Germany
Email: {farid.oubbati, mathis.richter, gregor.schoener}@ini.rub.de

***Abstract*— Humans excel at coordinating complex actions with the environment, most visibly in racket and ball sports, but pervasive in most other human activity as well. This challenge has been taken up by roboticists in demonstrations of robotic catching, juggling, and ball playing. We address how multiple different movements may be timed in relation to sensory events, but also organized sequentially. The generation and organization of the movements is coupled to online sensory information so that the system adjusts to perturbations both by varying movement parameters and by activating or deactivating different movement components. The approach builds on oscillator and neural dynamics models of human timed action. We formulate the approach in a simple task in which a robot arm keeps hitting a ball back up an inclined plane. The reactions to different kinds of perturbations are demonstrated.**

I. Introduction

Although they come naturally to us, activities like playing table tennis are complex and challenging. Within split-seconds we must solve and coordinate many tasks that include detecting the ball, estimating its future trajectory, initiating a movement that will hit the ball at the right time with the right velocity, direction, and orientation of the racket, and returning the racket while tracking the ball to get ready for the next interception. These actions are continuously chained together and flexibly tuned to the environment, both to respond to abrupt perturbations (e.g., to still reach a ball that was reflected by the net) and to adapt to changing conditions (e.g., to adjust to different players or rackets).

Roboticists have studied problems of this kind because they exemplify core elements of autonomous action, in particular, the problem of timing a robot's action to external events. Broadly speaking, there are three strands of such work. (1) Specific robotic solutions to interception problems [1], [2], [3], [4] are often very fast and accurate, but use specific algorithms that are not easily generalized to generic timing tasks. Typically, the demonstrations only address a single timed action such as a single catch. Repeated timed actions are organized by algorithms that fail, however, to provide flexibility in the face of perturbations or change. (2) Approaches based on learning movement primitives from human demonstration are adaptive in the sense that they learn from a given demonstrator [5]. During production a learned primitive is selected or several are blended together [6], [7], [8]. The models produce periodic repetitions of timed acts, although the flexible updating to changing sensory input and the reorganization of action sequences in response to perturbations are not typically addressed. (3) Periodic timed motor acts may be generated from nonlinear oscillators into which an effector system is coupled [9], [10]. Such systems may repeat timed actions that can be synchronized with sensed events, but are limited with respect to the complexity and heterogeneity of the timed actions.

Missing from this literature is a systematic approach to the organization of timed actions in which both the individual timed movement components as well as the sequencing mechanism are tied to changing sensory information. In this paper, we outline a dynamical model that takes a first step into this direction. We focus on how different discrete timed movements (here, moving a racket to intercept an approaching ball, hitting the ball, and returning the racket to a reference line) can be organized into sequences and coordinated flexibly based on perceptual input.

We build on earlier work that models timed human movement [11] based on nonlinear neural oscillators [12]. In that work, discrete motor acts were generated from oscillators that were activated and deactivated based on sensory input [13], so that they were timed reliably relative to a sensed event while remaining coupled to the time varying sensory input. Previous robotic versions [14], [15] used this principle to organize initiation, updating, and termination of a single timed action. Here, we show how a set of different timed actions can be organized and coordinated. This requires an approach to behavioral organization that is sensitive to timing. We formulate the approach in terms of a specific, simple task in which a robot arm holding a racket keeps a ball on an inclined plane by hitting it back up whenever the ball returns to the bottom.

The authors acknowledge the financial support of the European Union Seventh Framework Programme FP7-ICT-2009-6 under Grant Agreement no. 270247—NeuralDynamics. This work reflects only the authors' views; the EC is not liable for any use that may be made of the information contained herein.

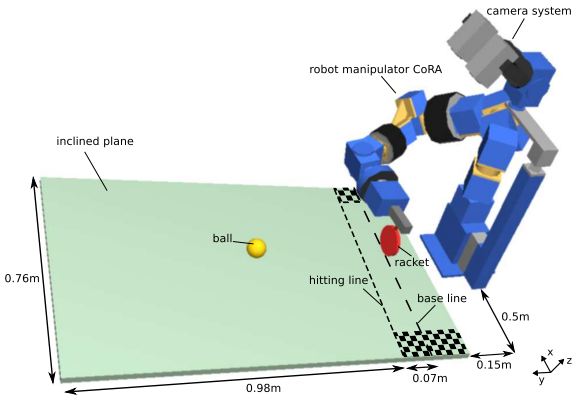

Fig. 1. Diagram of the experimental setup of the ball hitting task.

II. Robotic Scenario

Our robotic scenario consists of a simplified ball hitting task, in which a robot continuously hits a ball that is rolling down an inclined plane (see Fig. 1). We use a robot equipped with an anthropomorphic arm with eight degrees-of-freedom (DoF) and a vision system, which is used to track the ball and predict its trajectory. The robotic arm holds a small table tennis racket (10.5 cm in diameter) and must hit the ball with it to drive it back up the inclined plane. The region in which the robot arm can hit the ball is constrained by safety limits on both sides at the bottom of the inclined plane (marked with a checkerboard pattern in Fig. 1). We use a colored rubber ball with a radius of 3 cm and weight of 66 g. The arm moves forward to a virtual hitting line at the bottom of the inclined plane and hits the ball as it is crossing the line. After the ball is hit, the arm moves back to a virtual base line, ready to initiate another hitting movement. Ideally, the robot drives the ball back up the inclined plane with every hit and thus keeps it in play at all times.

This robotic scenario captures the essential problems we address in the paper: The task consists of different phases, all of which must be organized and coordinated based on input from the camera system. The movements have to be parametrized to be precise both in space and time to drive the ball back up the inclined plane in an angle that makes the next hit possible, if not easier. The scenario enables us to easily introduce different types of perturbations and analyze how the model deals with them.

III. Components of Timed Movement

A. Timed movements

Trying to hit an oncoming ball with a racket requires at least the following from the movements. (1) The movements need to be precise in space and time, such that the racket hits the target at a specific point in space and time. (2) The movements need to be able to adapt to changes in the target. (3) The movements must be switched on and off.

To fulfill these requirements, all movements within our model are controlled by a dynamical system that consists of different classes of behaviors. In the *movement behaviors*, a limit cycle oscillator controls a movement variable (here, the x- and y-position of the end-effector and its azimuth orientation) and executes a ballistic movement within a given time. In between movements, the *postural behaviors* stabilizes the movement variable at its current position.

The dynamical system for a movement is of the form

$$\tau \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = -c_{\text{post}}\, a \begin{pmatrix} x - x_{\text{post}} \\ y \end{pmatrix} + c_{\text{move}}\, H(x,y) + \eta, \quad (1)$$

where x is a timing variable that determines the end-effector trajectory, y is an auxiliary variable, and η is a noise term. The weights $c_{\text{post}}, c_{\text{move}} \in [0,1]$ can switch on the influence of the postural behaviors and movement behaviors, respectively. In the postural behaviors, x relaxes to the fixed point x_{post}, where the time constant τ and the constant $a > 0$ influence the relaxation time. In the movement behaviors, the change rate of x is determined by the Hopf oscillator

$$H(x,y) = \begin{pmatrix} \lambda & -\omega \\ \omega & \lambda \end{pmatrix} \begin{pmatrix} x - r - x_{\text{init}} \\ y \end{pmatrix} - \left((x - r - x_{\text{init}})^2 + y^2\right) \begin{pmatrix} x - r - x_{\text{init}} \\ y \end{pmatrix}, \quad (2)$$

where x_{init} is the value of x at the beginning of the movement. $r = \frac{1}{2}(x_{\text{target}} - x_{\text{init}})$ is the radius of the oscillator and x_{target} is the target position of the movement; $\lambda = r^2$ is derived from the radius. Finally, $\omega = \frac{2\pi}{T}$ is the angular frequency and T the cycle time of the oscillator. Note that Eq. 2 requires that the value of x_{init} be memorized. In our dynamical approach, we model this by

$$\dot{x}_{\text{init}} = c_{\text{mem}}\, a (x_{\text{init}} - x_{\text{real}}), \quad (3)$$

in which the memory x_{init} is updated to the current value of x_{real}. The initial value of the radius of the oscillator, r_{init}, is memorized by a similar dynamical system. This *update behavior* can be activated or deactivated using the weight $c_{\text{mem}} \in [0,1]$.

1) Adaptation of movement time: With the model described so far, we are able to generate ballistic movements that reach a target x_{target} in a specified time. However, the target position and the movement time are only predictions based on noisy sensory data. Even if the sensors were perfect, the oncoming ball could be perturbed and change its course. While the equations deal with changing target positions, the cycle time T of the oscillator needs to be adapted to accelerate or decelerate the movement. We update the cycle time to satisfy the ratio

$$\frac{T}{2 r_{\text{init}}} = \frac{t_{\text{tim}}}{d}, \quad (4)$$

where r_{init} is the radius of the oscillator at the beginning of the movement, t_{tim} is the time that remains for the

movement to be executed, and $d = x_{\text{target}} - x_{\text{real}}$ is the remaining distance.

To summarize, a single timed movement consists of three separate behaviors: the postural, movement, and update behavior. In order to function properly, these behaviors must be activated and deactivated in the correct sequence: the initial position must be memorized before starting to move and the movement has to suppress the postural behavior. The necessity of organizing behaviors in time becomes even more apparent when building entire architectures based on discrete behaviors.

The framework for behavioral organization is based on DFT, which we now briefly review.

B. Dynamic Field Theory

Dynamic Field Theory (DFT) [16] is a neural variant of the attractor dynamics approach. We use it here as an integrating framework between the low level sensory-motor streams of the robot and the higher level cognitive functions of the model, for instance its perceptual representations and its organization of behaviors.

Within DFT, dynamic neural fields (DNFs) are used to represent neural activity patterns over continuous, metric feature dimensions (e.g., color or space). The activation pattern evolves in continuous time t, as described by the following dynamic equation, which can be traced back to Grossberg [17] and was analyzed by Amari [18]

$$\tau \dot{u}(x,t) = -u(x,t) + h + S(x,t) + \int \omega(x-x') f(u(x,t)) dx'. \quad (5)$$

In Eq. 5, $u(x,t)$ describes the activation of a DNF at feature location x and time t. Without external input $S(x,t)$, the activation will relax to the resting level $h < 0$ and the output of the DNF, given by the sigmoidal function $f(u(x,t))$, will be zero. With sufficient external input, the DNF will produce output as well as lateral interaction within the feature dimension. The type of interaction is governed by the interaction kernel $\omega(\Delta x)$ and comprises local excitation and global or mid-range inhibition, promoting the formation of localized peaks of activation. Within DFT, such peaks are the units of representation for motor parameters, perceptual items, and memory items.

A zero-dimensional DNF is a dynamical node that represents a discrete instance of a percept or behavior.

C. Behavioral organization

A framework for behavioral organization based on DFT, previously introduced and implemented on a humanoid robot in a grasping task [19], is extended to flexibly organize timed behaviors.

1) Elementary behaviors: Within DFT, the behaviors that are organized are *elementary behaviors* (EB). EBs consist of two parts, an *intention* and a *condition of satisfaction (CoS)*, each of which is represented by a dynamical node and a dynamic neural field (DNF) (see Fig. 2). While the intention node simply determines whether the EB is active or inactive, the intention field describes the EB's connection to the world. For instance, the intention field of an EB 'move arm' would represent desired movement parameters of the arm (e.g., the target position) and would be connected to its motors.

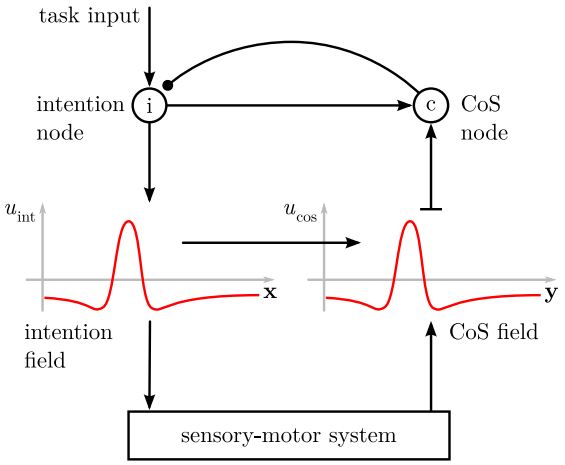

Fig. 2. Elementary behavior (EB) in Dynamic Field Theory. Each EB consists of two parts: the *intention* represents the desired change of the EB in the world, while the *condition of satisfaction (CoS)* represents the sensory signal expected for the successful completion of the EB.

The CoS field of an EB receives input from the intention field, describing the desired outcome of the EB (e.g., the end-effector of the arm at the target position). Additionally, the CoS field receives input from the sensory system, describing the current state of the EB (e.g., the current position of the end-effector). If the two inputs overlap, a peak forms in the CoS field, signaling the successful completion of the EB. This peak activates the CoS node, which in turn inhibits the intention node, switching off the EB. Explicitly modeling the beginning and end of an EB in this way allows us to close the gap between discrete actions and the continuous sensory-motor streams they are connected to.

2) Generating sequences: By default, all EBs relevant for a task are activated at the same time. The sequential organization of EBs derives from constraints that are represented by dynamic coupling terms and are defined within pairs of EBs. A *precondition* constraint prevents a first EB (e.g., here, the update EB) from becoming active until a second EB (e.g., here, the movement EB) is completed. The constraint is represented by a precondition node, a dynamical node that is activated by task input and inhibits the intention node of the second EB. As soon as the first EB is finished, its CoS node will activate and inhibit the precondition node, releasing the intention node of the second EB from inhibition and thereby allowing its execution.

A *suppression* constraint between two other EBs prevents one of them (e.g., here, the postural EB) from becoming active while the other (e.g., here, the move-

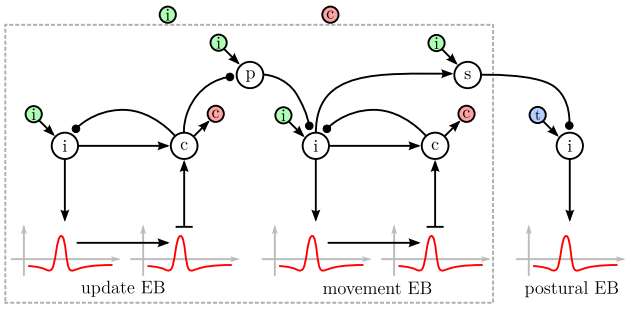

Fig. 3. A movement module as used in our architecture to control timed movements. The two leftmost EBs (in the dashed box) can be activated and deactivated by EBs on a higher level (using the colored nodes as input and output), while the postural EB is always active by task input (blue node). The nodes labeled 'p' and 's' represent the precondition and suppression node, respectively.

ment EB) is active at the same time. This constraint is represented by a suppression node, a dynamical node activated by task input and simultaneous input from the intention node of the first EB. When activated, the suppression node inhibits the intention node of the other EB. A second suppression constraint can be added to create a *competition*—a bidirectional suppression constraint, where the two EBs suppress each other.

D. Movement module

With these concepts of behavioral organization, we can now organize the three (elementary) behaviors of a single timed movement (see Section III-A) into a *movement module*. As shown in Fig. 3, each of the EBs is represented by an intention node and field, and condition of satisfaction (CoS) node and field. The postural EB does not have a CoS because it is always active unless suppressed. Additionally, we have defined behavioral constraints that generate a meaningful sequence of these three EBs: a suppression node deactivates the postural EB whenever the movement EB is active and a precondition node makes sure that the update EB is active before the movement is initiated. The output (i.e., the sigmoided activation) of the intention node of each EB is then used as control variables (i.e., $c_{\text{post}}, c_{\text{move}}, c_{\text{mem}}$ of Eq. 1 and 3), thereby controlling the influence of each EB on the motors of the robot.

As long as the movement module is deactivated, the postural EB remains active and holds the end-effector at its current position. When the module is activated, the excitatory top-down input reaches both the update EB and the movement EB. Due to the precondition constraint, the update EB is executed first and updates the initial position of the movement, leading to the CoS node of the update EB to become active. The latter inhibits the precondition node, which in turn releases the movement EB from inhibition. As soon as that EB turns on, the suppression node deactivates the posture EB and the ballistic movement is executed. Once it is finished, the posture EB is no longer suppressed and reactivates.

IV. Architecture

A. Perception and motor systems

The main perceptual input of the architecture (see Fig. 4) is based on the signal of a single, stationary camera that monitors the inclined plane. We estimate the position of the ball based on a color segmentation and use a Kalman estimator to predict the point, x_{hp}, at which the ball will cross the hitting line and the time, t_{tim}, it will take the ball to reach that point. Due to the simplicity of the estimator, we only get those predictions when the ball is rolling down the inclined plane. The prediction of the hitting point x_{hp} is represented by a peak of activation in a dynamic neural field (the 'perceptual field' in Fig. 4) that is defined over the width of the inclined plane (i.e., its x-axis). Thus, the field only has a peak if the perceptual input currently yields a prediction of where the ball will cross the hitting line and if that point lies within the safety limits of the robot (see Fig. 1).

On the motor end of the architecture, our model controls the position and orientation of the end-effector in space, where the end-effector is defined as the center of the table tennis racket. That control signal is converted to velocities for the eight joints of the robot arm using a closed form solution for the inverse kinematics.

B. Hierarchical structure and behavioral organization

Fig. 4 shows that the architecture has two hierarchical levels of elementary behaviors. The lower level consists of three movement modules, controlling the position of the end-effector along the x-axis and y-axis of the inclined plane, and the azimuth orientation ϕ of the end-effector, respectively. These movement modules can be activated and reused by the four EBs on the higher level

- *Move to ball*: This EB moves the end-effector toward the point x_{hp}, where the ball trajectory will cross the hitting line. The EB activates the movement modules controlling both the x- and y-position of the end-effector.
- *Move to base line*: Moves the end-effector back toward the base line of the inclined plane but controls the end-effector position along the y-axis only.
- *Move hand forward*: This is the actual hitting movement, which involves only the hand segment of the arm and controls the azimuth orientation ϕ of the racket.
- *Move hand backward*: Moves the hand segment of the arm back to a reference angle. This EB also only controls ϕ.

The architecture is autonomous in that it organizes and coordinates its own behaviors based on sensory input and the prediction of the ball trajectory derived thereof. The EBs "move to ball" and "move hand forward" can only be activated when a prediction is available for the hitting point of the ball. That means, there is a precondition constraint between the perceptual field,

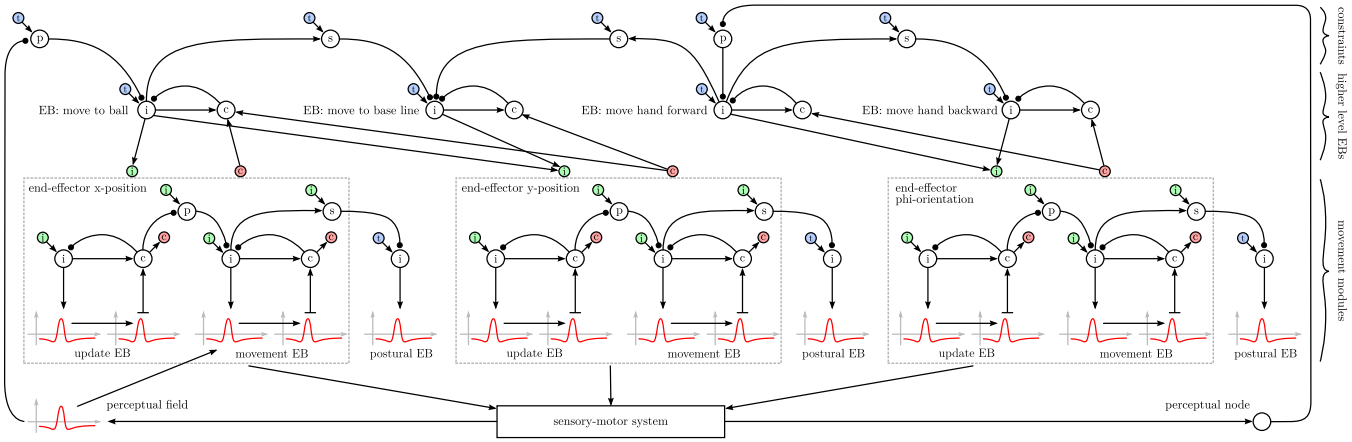

Fig. 4. Architecture that controls the robotic arm and organizes its behaviors in the ball hitting task.

representing the prediction, and the two EBs (see Fig. 4). Each of the EBs suppresses the EB for the reciprocal movement (e.g., "move hand forward" suppresses "move hand backward"). Thus, without a prediction, the default behavior of the robot is to return to the base line and retract its hand in anticipation of the next ball. Once the arm has returned, the postural behaviors are activated to keep the arm at its current position. To reduce the distance that the arm has to travel toward the ball along the x-axis in the next timed movement the end-effector tracks the position of the ball until the next prediction of x_{hp} is available. As soon as that happens, the EB "move hand forward" is disinhibited. A second precondition is lifted from the EB "move hand forward" as soon as the ball has reached a certain distance threshold and initiates the hit.

C. Racket orientation and timing during hitting

In our scenario, the ball is the easier to hit the straighter it rolls down the inclined plane and the earlier the prediction of its hitting point is available. Thus, during the hit, we control the racket to drive the ball up the inclined plane as straight and with as much speed as possible.

We assume that upon impact, the ball makes an ideal elastic collision with the racket. From the incoming velocity vector of the ball we continuously compute the azimuth orientation the racket must have during the hit in order to reflect the ball straight up the inclined plane. However, since the velocity vector of the ball changes over time, the desired racket orientation—and thus our entire movement plan—changes as well. To adapt to that change, we adjust the cycle time of the movement as described in Eq. 4. This gives us a timed movement that correctly orients the racket during the hit.

We maximize the speed of the hitting movement by controlling its initiation time. The hitting movement initiates when the time-to-impact falls below a variable threshold. We tune that threshold so that the velocity required to execute the movement stays just below a maximum velocity allowed for the end-effector due to hardware limitations.

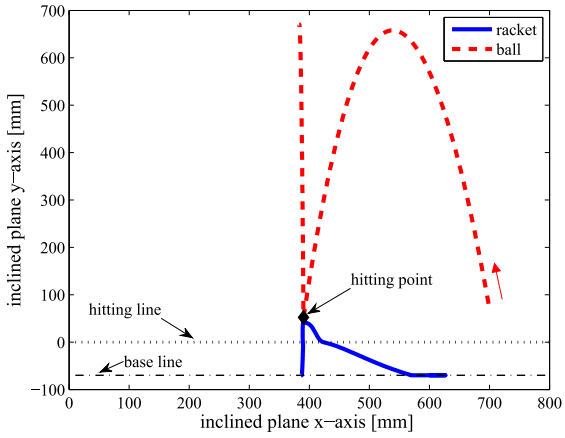

Fig. 5. Trajectories of the ball and the racket for a successful hit.

V. Results

We present results from a physically realistic Matlab simulation as well as preliminary results from a hardware implementation that illustrate the flexibility of our approach to organize timed movements.

A. Simulation results

The simulation results consist of four experiments in which we perturb the trajectory of the ball in different phases of the robot's movement. In an additional experiment, we gathered statistical data about the performance of the robot.

1) Successfully hitting the ball: The first scenario simply demonstrates that the robot is able to hit the ball and drive it back up the inclined plane. Fig. 5 shows the trajectories of both the ball and the racket when viewed from above the inclined plane.

We will now use this experiment to explain what is happening within our model during a hitting movement.

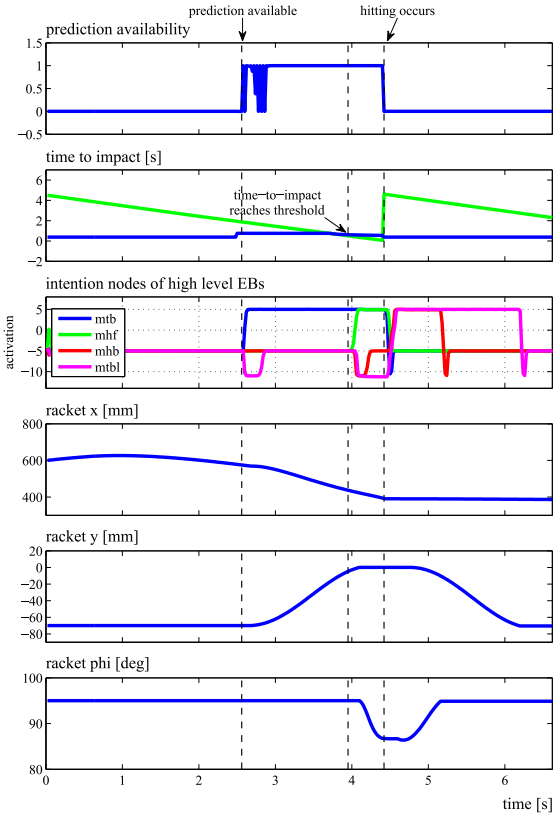

Fig. 6. Time courses of meaningful variables of the architecture during a successful hit. From top to bottom, the plots show (1) whether a prediction for the ball hitting point x_{hp} is available, (2) the time to impact, (3), the activation of the intention nodes of the high level EBs, (4,5) the position of the racket along the x- and y-axis, and (6) the azimuth orientation of the racket.

You can follow along using the time courses of some of the meaningful variables of the model shown in Fig. 6. At $t = 0\,\mathrm{s}$, the ball is launched upwards from the bottom of the inclined plane. It reaches the highest point and starts rolling back down the plane at $t \approx 2.56\,\mathrm{s}$, at which time the vision system provides a prediction of the hitting point and the time-to-impact, forming a peak in the perceptual field. The intention node of the EB 'move to ball' (abbreviated with 'mtb' in Fig. 6) turns on and initiates a timed movement that drives the end-effector toward the predicted hitting point. As the ball approaches the hitting line and the estimated time-to-impact falls below a variable threshold value (at $t \approx 3.95\,\mathrm{s}$), the EB 'move hand forward' (mhf) gets activated. It starts a timed movement of the racket orientation ϕ, leading to a hit of the ball ($t \approx 4.42\,\mathrm{s}$). The hit drives the ball back up the inclined plane, removing the prediction of where the ball will cross the hitting line (i.e., the peak in the perceptual field falls away). The intention nodes of the EBs 'move hand backward' (mhb) and 'move to base line' (mtbl) switch on and initiate movements that drive

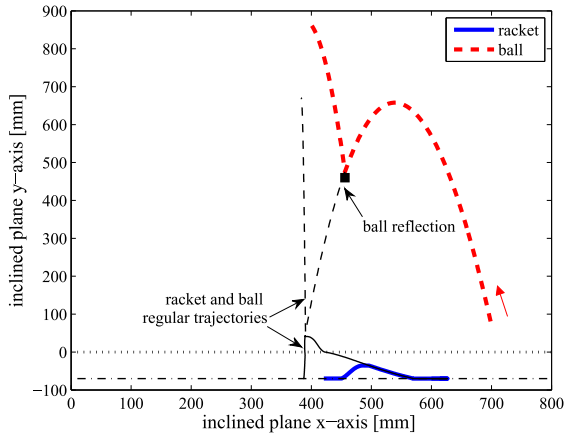

Fig. 7. Trajectory of the racket when the ball is reflected during the racket movement. As before, the colored lines show the trajectories of the ball and racket. The thin black lines show the trajectories had the ball not been reflected.

the racket orientation ϕ back to the initial orientation and the position of the end-effector back to the base line (along the y-axis), respectively. At the same time, the end-effector starts tracking the ball along the x-axis.

2) Ball reflected during movement: This experiment demonstrates how the model reacts when the prediction of the hitting point disappears while the end-effector is moving toward the ball. To probe this in the experiment, the ball is reflected and suddenly starts moving upward on the inclined plane. Fig. 7 shows that the model autonomously reacts to this unexpected change by initiating a fixed time movement back to the base line. The end-effector waits there, ready to initiate the next hitting movement, while tracking the ball position horizontally.

3) Hitting movement re-initiation: After a hit, a timed movement usually brings the end-effector to the base line to get ready for the next hitting action. In this experiment, the ball is deliberately hit too weakly, so that it quickly returns and a new prediction of its hitting point is available before the end-effector reaches the base line (see Fig. 8). The model initiates a hitting movement while the end-effector is still on its way to the base line. The robot is able to hit the ball again.

4) Ball deviated during movement: As described in Section III, the cycle time of the timed movements is adapted online to changes in the predicted target position. This experiment demonstrates this adaptation by deviating the ball trajectory while the end-effector is moving toward the predicted hitting point of the ball. Fig. 9 shows the trajectory of both the ball and the end-effector alongside the trajectories they would have had, had the path of the ball not been altered. It also shows that the robot is able to adapt to the deviation and accelerate the movement to successfully hit the ball.

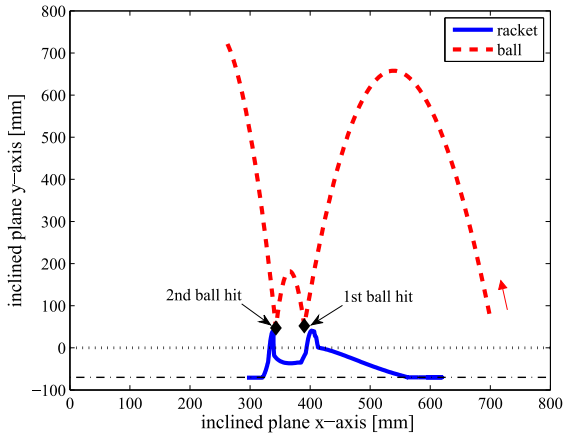

Fig. 8. A hitting movement is re-initiated while the end-effector is still moving to the base line.

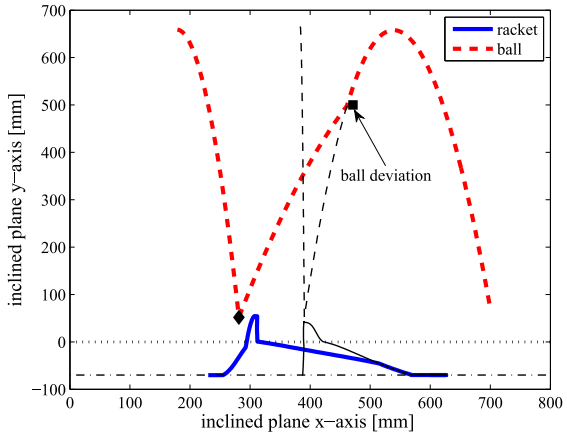

Fig. 9. Trajectory of the racket when the ball is deviated to the left during the racket movement. The thin black lines show the trajectories of the ball and racket had the ball not been deviated.

5) Statistical data: Statical data were gathered during a simulated experiment in which the robot continuously drove the ball up the inclined plane for as long as possible. The experiment consisted of 1000 trials, where in each trial the robot hits the ball until it fails. After it misses a ball, the next trial is initiated by introducing a new ball on the inclined plane with a random speed and launching angle.

If the ball crosses the hitting line within the safety zones that the robot arm cannot reach (see Fig. 1), this is not counted as a failure. A new ball is introduced without resetting the hitting counter.

The statistical data was gathered for inclinations of 5° and 10° of the plane and is shown in Table I. The success rate is computed as the number of successful hits divided by the total number of hits (including fails). The other measures are the mean, standard deviation and median of successive hits. For a plane inclination of 5°, the mean deviation between the position of the racket and the ball

TABLE I

PERFORMANCE OF THE SIMULATED ROBOT ON SUCCESSIVE BALL HITS WITH TWO DIFFERENT PLANE INCLINATIONS.

Plane incl. (°)	Success rate (%)	Mean (Hits)	Standard dev. (Hits)	Median (Hits)
5	95.442	20.938	34.781	10
10	92.432	12.214	11.305	9

TABLE II

PERFORMANCE OF THE REAL ROBOT ON SUCCESSIVE BALL HITS.

Plane incl. (°)	Success rate (%)	Mean (Hits)	Standard dev. (Hits)	Median (Hits)
2	95.192	19.8	16.825	19

at the hitting moment is 21.2 mm, while for failed hits, the mean deviation is 68 mm. For the plane inclination of 10°, the mean deviation is 18 mm during successful hits and 134.9 mm during failed hits.

B. Hardware results

To evaluate the hardware implementation, we repeated the last experiment shown for the simulation, in which the robot continuously drives the ball up the inclined plane for as long as possible. Table II shows the statistical data gathered during 10 trials. As in simulation, the ball is reintroduced after the robot misses it. If the ball crosses the hitting line within the safety zones, a new ball is reintroduced without resetting the hitting counter. The limitation in the inclination angle is due to the inherent speed limits of the robotic hardware. The maximum number of consecutive hits was 54. Without ignoring balls that pass through the safety zones, the maximum was 25. Fig. 10 shows a photo of the robotic arm hitting the ball.

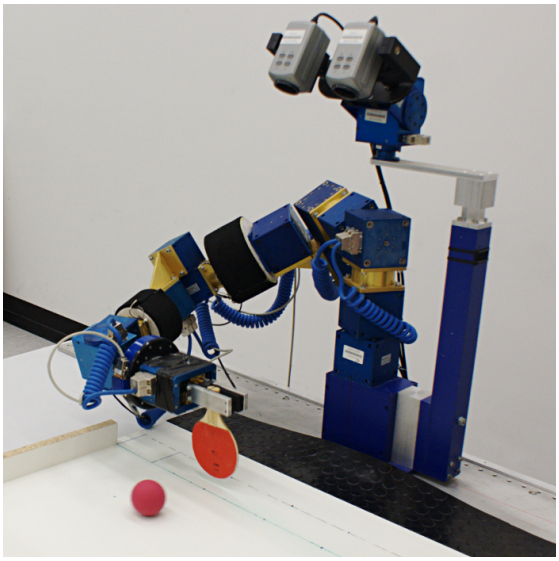

Fig. 10. Photo of the robotic arm just before hitting the ball.

VI. Conclusions

The architecture developed here integrates the generation and the organization of timed movements through coupled limit cycle oscillators and neural attractor dynamics that are tied to online sensory information. As a result, the system is ready to abort running behaviors as well as to initiate or re-initiate inactive behaviors at any time in response to external perturbations. The system may also update movement parameters such as amplitude and movement time on the fly when such perturbations shift the timing and spatial constraints. Here we demonstrated the system in a ball hitting scenario that entails coordinating multiple different actions, while remaining relatively simple. Because the approach is consistent with a body of work on neural dynamics architectures for cognitive robotics [20], it enables integrating timing constraints into cognitive robots more broadly.

References

[1] W. Hong and J.-J. E. Slotine, "Experiments in hand-eye coordination using active vision," in *Experimental Robotics IV*, vol. 223, pp. 130–139, Springer, 1997.

[2] T. Senoo, A. Namiki, and M. Ishikawa, "Ball control in high-speed batting motion using hybrid trajectory generator," in *IEEE International Conference on Robotics and Automation*, pp. 1762–1767, IEEE Press, 2006.

[3] B. Bäuml, T. Wimböck, and G. Hirzinger, "Kinematically optimal catching a flying ball with a hand-arm-system," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2592–2599, 2010.

[4] H. Li, H. Wu, L. Lou, K. Kühnlenz, and O. Ravn, "Ping-pong robotics with high-speed vision system," in *12th International Conference on Control Automation Robotics & Vision*, 2012.

[5] S. Schaal, A. Ijspeert, and A. Billard, "Computational approaches to motor learning by imitation," *Philosophical Transactions of the Royal Society (London) Series B*, vol. 358, pp. 537–547, 2003.

[6] S. Kim, E. Gribovskaya, and A. G. Billard, "Learning motion dynamics to catch a moving object," in *10th IEEE-RAS International Conference on Humanoid Robots*, pp. 106–111, IEEE Press, 2010.

[7] K. Mülling, J. Kober, and J. Peters, "A biomimetic approach to robot table tennis," in *International Conference on Intelligent Robots and Systems*, pp. 1921–1926, IEEE Press, 2010.

[8] J. Kober, M. Glisson, and M. Mistry, "Playing catch and juggling with a humanoid robot," in *IEEE-RAS International Conference on Humanoid Robots*, IEEE Press, 2012.

[9] M. Buehler, D. E. Koditschek, and P. J. Kindlmann, "Planning and control of robotic juggling and catching tasks," *The International Journal of Robotics Research*, vol. 13, no. 2, pp. 101–118, 1994.

[10] A. Nakashima, Y. Sugiyama, and Y. Hayakawa, "Paddle juggling of one ball by robot manipulator with visual servo," in *Proceedings of the International Conference on Robotics and Automation*, IEEE Press, 2006.

[11] W. H. Warren, "The dynamics of perception and action," *Psychological Review*, vol. 113, no. 1, pp. 358–389, 2006.

[12] G. Schöner and J. A. S. Kelso, "Dynamic pattern generation in behavioral and neural systems," *Science*, vol. 239, pp. 1513–1520, 1988.

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

[14] G. Schöner and C. Santos, "Control of movement time and sequential action through attractor dynamics: A simulation study demonstrating object interception and coordination," in *Proceedings of the 9th Intelligent Symposium On Intelligent Robotic Systems*, pp. 15–24, 2001.

[15] M. Tuma, I. Iossifidis, and G. Schöner, "Temporal stabilization of discrete movement in variable environments: an attractor dynamics approach," in *IEEE International Conference on Robotics and Automation*, pp. 863–868, IEEE Press, 2009.

[16] G. Schöner, *Dynamical Systems Approaches to Cognition*, ch. Dynamical, pp. 101–126. Cambridge, UK: Cambridge University Press, 2008.

[17] S. Grossberg, "A theory of human memory: self-organization and performance of sensory-motor codes, maps, and plans," in *Progress in Theoretical Biology, Vol. 5* (R. Rosen and F. Snell, eds.), pp. 500–639, 1978.

[18] S.-i. Amari, "Dynamics of pattern formation in lateral-inhibition type neural fields," *Biological Cybernetics*, vol. 27, no. 2, pp. 77–87, 1977.

[19] M. Richter, Y. Sandamirskaya, and G. Schöner, "A robotic architecture for action selection and behavioral organization inspired by human cognition," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2457–2464, IEEE Press, 2012.

[20] S. K. U. Zibner, C. Faubel, I. Iossifidis, and G. Schöner, "Dynamic neural fields as building blocks of a cortex-inspired architecture for robotic scene representation," *IEEE Transactions on Autononous Mental Development*, vol. 3, no. 1, pp. 74–91, 2011.