Proceedings of the 45th IEEE Conference on Decision & Control
Manchester Grand Hyatt Hotel
San Diego, CA, USA, December 13-15, 2006

Using Dynamic Optimization for Reproducing the Chameleon Visual System

Ofir Avni*, Francesco Borrelli†, Gadi Katzir‡, Ehud Rivlin* and Hector Rotstein§
*Department of Computer Science, Technion - Israel Institute of Technology, Haifa, Israel
†Università del Sannio, Benevento, Italy
‡Department of Biology, Oranim - University of Haifa, Tivon, Israel
§Armament Development Authority and
Department of Electrical Engineering, Technion - Israel Institute of Technology, Haifa, Israel

Abstract— **This paper presents a method for robust target tracking with a set of independent pan-tilt cameras which combines smooth pursuit and saccades. The approach is systematic and optimization-based, and was inspired by our understanding of the chameleon visual system. Explicit robust and minimum-time Model Predictive Control (MPC) theory is used for the design of the tracking control law. Experimental validation of the approach on a robotic chameleon head composed of two independent pan-tilt cameras is presented.**

I. Introduction

Environment visual scanning (EVS) is a critical task for living creatures and artificial systems alike. EVS refers to the process of visual exploration of the environment used to identify relevant events and detect threats or targets. In nature, solutions to EVS are diverse and range from: *(i)* non-moving eyes on top of a movable neck as in the barn owl, through *(ii)* coupled movements of the eyes as in humans, to *(iii)* independent eye movements as seen in chameleons. Overall, one can identify three models which represent different levels of eye vs. neck movements and correspond to the three examples mentioned above: the owl-like, the human-like and the chameleon-like models. In the world of robotics, common solutions are based on the first two models. For example, a number of systems have been implemented using two cameras that are either fixed or verged [1]–[3]. These two models have a number of important advantages in terms of facilitating visual calculations such as estimating depth from stereo. Yet, by constraining the relative motion of the cameras they sacrifice search efficiency due to a large overlap between the views of the two cameras.

The chameleons are probably the best known example for independent eye movements. Eye movements in chameleons are considered unique. They are laterally positioned and can move independently, each over a wide range, while scanning different sections of the environment. Independent eye movements are seen also in fishes. A well studied example is that of the sandlance which has an optical system and eye movements similar to the chameleon [4]. Scanning in chameleons is performed by saccades - i.e. large, independent and rapid movements of the eye from one location to another [5]. The range of movement of each eye is almost $180°$ in the pan axis and about $90°$ in the tilt axis. Recent studies of this manuscript' authors indicate that the global scanning strategy of the chameleon is based on a "negative correlation" principle. More specifically, if one eye scans forward, then with a high probability, the other will point backwards.

Motivated by this fact and the observations mentioned above, the objective of our research is to develop, model and control a robot head based on the principles that govern the chameleon eyes movements, without imposing a-priori constraints on the relative motion of the cameras. Figure 1 depicts our system as compared to a chameleon. It is assumed that the vision system has two main goals:

1) Scanning the environment autonomously while searching for targets or threats.
2) Acquiring and tracking targets or threats once they are detected.

Because of lack of space, *this paper focuses only the second goal*: active target tracking. Nevertheless, both goals are important in order to understand and reproduce the chameleon visual system. The reader is referred to [6], [7] for a detailed discussion on the topic and for a novel scanning algorithm which has a remarkable resemblance with the one of the chameleon. Also, it is worth mentioning that in real-life systems the oculomotor system has additional objectives related to the two mentioned above; an obvious example is range estimation to the prey, which is required for a successful capture. This paper does not dwell with such additional objectives as well as with the image processing tasks, i.e., with extracting information and features from the optical images.

In general, a good active target tracking can be achieved by combining *smooth pursuit* and *saccades* movements [8], [9]. In smooth pursuit, the target is tracked continuously with the camera motion controlled so that the image of the target remains inside a predefined window. Saccades are triggered either by large tracking errors (including the case where the image of the target exits the tracking window) or by the request of moving the camera to a new region of attention. Usually, the purpose of saccades is to quickly reorient the cameras in order to allow a smooth pursuit. The smooth pursuit/saccades scheme is a natural consequence of the foveated structure of biological eyes found in many creatures in nature including the chameleon. As shown by [8], [9], foveated vision and hence smooth pursuit/saccades can be explained in terms of optimal control theory. As a continuation of that work, Model Predictive Control is

1-4244-0171-2/06/$20.00 ©2006 IEEE.

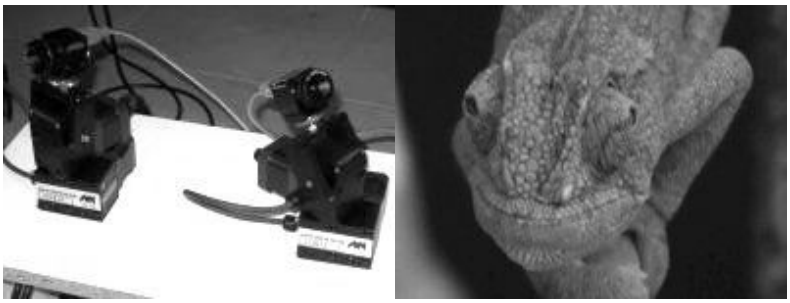


Fig. 1. The Robotic Head and a Chameleon Head

proposed in this paper as a more suited approach to design each control loop and their interaction. *Model Predictive Control* (MPC) is a control technique that uses the *model* of the plant to *predict* the future evolution of the system [10]. Based on this prediction, at each time step t a certain performance index is optimized under operating constraints with respect to a sequence of future input moves. The first of such optimal moves is the *control* action applied to the plant at time t. At time $t+1$, a new optimization is solved over a shifted prediction horizon. This leads to a feedback control policy.

In the past, MPC was limited to relatively "slow" processes as found in the chemical industry, since solving an optimization problem in real-time is computationally expensive. Recent developments have shown that it is possible to solve the optimization problem off-line as a function of all feasible states. In particular, for linear and piecewise-linear systems, the feedback solution of an MPC scheme with a linear or quadratic performance index and linear constraints is shown to be a piecewise linear function of the states [11], [12]. Therefore, MPC on-line implementation resorts to a simple lookup-table evaluation and allows the use of MPC in "fast", real-time systems. The interested reader is referred to the manual of the Multi-Parametric Toolbox [13] for a quick introduction to the topic. A detailed description of latest results on the topic can be found in [14].

The paper is organized as follows. In Section I-A we present in more details the principles that govern the eyes movements in chameleons. In Section II the smooth pursuit controller, the saccade controller and the way they are combined together are discussed. Section III contains experimental results of the proposed approach.

A. The Visual System of the Chameleon

Chameleons are arboreal lizards that eat mostly insects. They are rather slow and developed, as a compensation, a unique visual system combined with a specialized tongue in order to catch a prey. The chameleon's eyes are positioned laterally on the head and are able to move independently over a wide range. This enables the chameleon to quickly scan the environment while improving its camouflage, since only its eyes are moving.

The chameleon's eye has the basic structure of a simple chambered eye common to all vertebrates. The lens, unlike other vertebrates, has a negative power [15]. This results in a magnification of the image on the retina, which enables a more accurate measurement of image focus and probably supports more accurate distance measurement from accommodation cues. Before shooting the tongue, the eyes alternate between states of in-focus and out-of-focus [16] which may serve to perform distance estimation by focus/defocus algorithms.

As mentioned in the introduction, in order to scan the environment, chameleons performs large, independent saccadic movements of the eyes. Once a target is detected the head axis is directed towards it and both eyes fixate it. If the target starts to move the chameleon tracks it using a combination of head movements and eye movements [17]. In [17] the author highlights that while the saccadic movements are independent during the search for the prey, they are synchronous during the tracking of the prey. The chameleon is the only vertebrate known to switch from independent to synchronous saccades. Additional details can be found in [6], [7].

II. Target Tracking: Smooth Pursuit and Saccade

In this section we introduce the smooth pursuit controller, the saccade controller, and the method they are combined together. The smooth pursuit controller is a robust controller and relies on the assumption that the target model is excited by an unknown and bounded disturbance which brings it away from the center of a given tracking window. The saccade controller is a minimum-time optimal controller that takes the opposite approach: it assumes that the target movement is known, without any disturbances, and tries to center the target in the tracking window as fast as it can. We start by presenting the used dynamical model of a single pan-tilt head (Figure 2a).

A. Active Vision Model

We denote by ψ and θ the pan angle and the tilt angle, respectively, and by $\dot{\psi}$ and $\dot{\theta}$ the pan and tilt speeds, respectively. By collecting the model variables, the head state vector can be denoted by $x_p = \left[\psi, \dot{\psi}, \theta, \dot{\theta}\right]^T$. We use discrete-time linear time-invariant model with sampling time equal to the sampling time of the camera. With abuse of notation the same notation is used for the corresponding discrete time

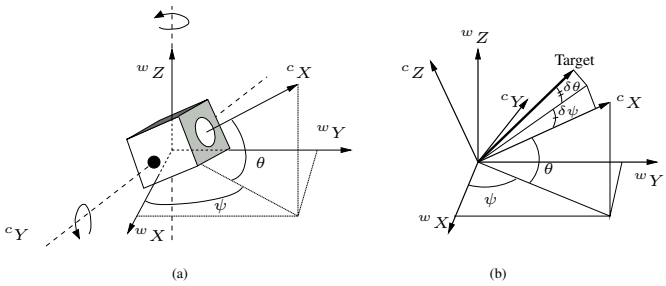


Fig. 2. Pan Tilt Model and Coordinate Systems - The axes marked with ^{w} define the world coordinate system. The axes marked with ^{c} define the coordinate system in the camera framework. The camera is pointing toward its X axis (and not to the standard Z axis). The axes corresponding to the image hight and width are the Z and Y axes respectively. The pan and tilt axes are shown in sub-figure (a). The measured tracking errors $\delta\psi$ and $\delta\theta$ are shown in sub-figure (b).

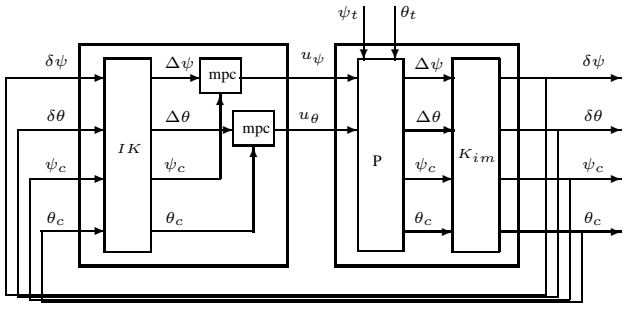


Fig. 3. System Model described in Section II-A- The large block on the left is the control. The large block to the right is the plant.

variables, $x_p = \left[\psi, \dot{\psi}, \theta, \dot{\theta}\right]^T$. The head dynamics, including the dynamics of the motors and their local controllers, are described by the model

$$x_p(k+1) = A_p x_p(k) + B_p u(k)$$

where A_p is a 4×4 matrix, B_p is a 4×2 matrix and u is the 2×1 input vector to the local pan-tilt controllers.

The target dynamics are modeled as a double integrator in the radial coordinate system. If A_t and B_t define the dynamics of double integrator, and $x_t = \left[\psi_t, \dot{\psi}_t, \theta_t, \dot{\theta}_t\right]^T$ is the target state, then the target dynamics can be written as

$$x_t(k+1) = A_t x_t(k) + B_t v(k)$$

where $v(k)$ is the acceleration of the target which is unknown to the cameras. We define the global tracking error as $\Delta x = x_t - x_p$, and we separate it into global tracking error in the pan axis, $\Delta\Psi = \Psi_t - \Psi_t$ and global tracking error in the tilt axis, $\Delta\Theta = \Theta_t - \Theta_p$, where $\Psi = \left[\psi, \dot{\psi}\right]^T$ and $\Theta = \left[\theta_t, \dot{\theta}_t\right]^T$.

We denote by $\delta\psi$ and $\delta\theta$ the measured tracking errors. It should be noticed that the measured tracking error are referred to the camera coordinate system (Figure 2b) which is different from the global coordinate system in our configuration. It should also be noticed that only the position tracking error is measured and that the tracking error speed is estimated from it. Given the measured tracking errors $\delta\psi$ and $\delta\theta$ and the current position of the camera ψ and θ, the target position in the world coordinate system can be calculated as

$$\begin{aligned} \psi_t &= -\arcsin\left({}^wT_z / \sqrt{{}^wT_x^2 + {}^wT_y^2}\right) \\ \theta_t &= \arctan\left({}^wT_y / {}^wT_x\right) \end{aligned}$$

where ${}^wT = [{}^wT_x, {}^wT_y, {}^wT_z]$ is a normal vector pointing toward the target in the world coordinate system. ${}^wT$ can be calculated from the line of sight in the camera coordinate system and by using ${}^cT$, through the equation:

$${}^wT = R_y(\theta) \cdot R_z(\psi)\, {}^cT,$$

where $R_O(\alpha)$ is the rotation matrix around the axis O, with α radians. The vector ${}^cT$ is calculated in a similar way using $\delta\psi$ and $\delta\theta$.

$${}^cT = R_y(\delta\theta) \cdot R_z(\psi) \cdot [1,0,0]^T.$$

The global tracking error can then be easily calculated by subtracting the current camera position, $\Delta\psi = \psi_t - \psi$ and $\Delta\theta = \theta_t - \theta$.

Clearly the global tracking errors $\Delta\psi$ and $\Delta\theta$ are not linearly dependent on the measured errors $\delta\psi$ and $\delta\theta$. In order to make use of a linear MPC, we preprocess the measured errors with a nonlinear static inversion. In particular, after measuring $\delta\psi$ and $\delta\theta$, the global tracking errors $\Delta\psi$ and $\Delta\theta$ are computed by a non-linear function (denoted IK). The MPC controller receives as inputs the latter tracking errors. Since a minimization of $\Delta\theta$ implies the minimization of $\delta\theta$, we obtain a linear MPC with the same final objective.

Figure 3 presents the system model. The large left block performs the control. The large block to the right is the plant. In the control block the IK non-linear function computes the tracking errors $\Delta\psi$ and $\Delta\theta$ from the measured tracking errors $\delta\psi$ and $\delta\theta$ and the current head position ψ_c and θ_c. In the plant, the block marked as P represents the local pan-tilt controllers and the pan-tilt engines. It receives the input from the MPC controllers and considers the target movements as a disturbance. The outputs of P are the tracking error $\Delta\psi$, $\Delta\theta$ and the head position. However, through image precessing one can measure only $\delta\psi$ and $\delta\theta$, outputs of the static non-linear function K_{im}.

In addition to the linearity of the problem, by using the global tracking error rather the measured ones, we gain the separability of the control problem. The pan and tilt controllers can be now separated since they are dynamically decoupled.

B. Smooth Pursuit Control

We combine the camera states and target states into the following LTI dynamical model:

$$\begin{bmatrix} x_p(k+1) \\ x_t(k+1) \end{bmatrix} = \begin{bmatrix} A_p & 0 \\ 0 & A_t \end{bmatrix} \begin{bmatrix} x_p(k) \\ x_t(k) \end{bmatrix} + \begin{bmatrix} B_p \\ 0 \end{bmatrix} u(k) + \begin{bmatrix} 0 \\ B_t \end{bmatrix} v(k)$$

We apply a linear state transformation by replacing the target states with the tracking error states:

$x = \left[x_p, \dot{x_p}, \Delta x, \dot{\Delta x}\right]$, where $\Delta x = x_t - x_p$ and $\dot{\Delta x} = \dot{x}_t - \dot{x_p}$. The resultant model will be compactly written as

$$x(k+1) = Ax(k) + Bu(k) + Ev(k) \tag{1}$$

The goal of the smooth pursuit controller is to maintain the target inside a predefined tracking window. The acceleration $v(k)$ that drives the target can be seen as a disturbance that competes against the controller goal. The smooth pursuit controller is designed by solving a min-max game where the disturbance is the opponent and the controller tries to minimize the worst-case performance index while satisfying the constraints for all the possible disturbance. State constraints will include the presence of the target within a pre-specified window and bounds on position, variation and acceleration of the head angles. We design a min-max MPC controller by solving the following optimization problem

$$\begin{aligned}
&\min_{u_k \in \mathbb{U}} && \left\{ \|Ru_k\|_p + \max_{v_k \in \mathbb{V}} \left\{ \left\|Qx_{k+1|k}\right\|_p + \dots \right.\right. \\
&s.t \quad x_k \in \mathbb{X} \ \ \forall v_k \in \mathbb{V} \\
&\qquad \min_{u_{k+N-1} \in \mathbb{U}} && \left\{ \|Ru_{k+j-1}\|_p + \right. \\
&\qquad s.t \quad x_{k+N-1} \in \mathbb{X} \ \ \forall v_{k+N-1} \in \mathbb{V} \\
&\left. \left. \left. + \max_{v_{k+N-1} \in \mathbb{V}} \left\{ \left\|Qx_{k+N|k}\right\|_p \right\} \right\} \right\} \right\}
\end{aligned} \tag{2}$$

where N is the prediction horizon, $Q \in \mathbb{R}^{n \times n}$ defines the weights on the states and $R \in \mathbb{R}^{m \times m}$ defines the weights on the control inputs. The vector $x_{k+j|k}$ denotes the predicted state at time $k+j$ starting from state $x_{k|k} = x(k)$ and applying the input sequence $u_k, \dots, u_{k+j-1}$, and the disturbances sequence $v_k, \dots, v_{k+j-1}$ to the model (1). The sets $\mathbb{X}$, $\mathbb{V}$, and $\mathbb{U}$ are polyhedral sets that define constraints on the states, disturbances and inputs. In particular, the set $\mathbb{X}$ constraints the target position to be in a certain window and the set $\mathbb{V}$ defines the maximum amplitude of the target acceleration.

Solving this optimization problem (2) in real-time might be challenging. In the case of a piecewise-linear performance index ($p = 1, \infty$), an off-line solution to problem (2) can be computed as a function of the current state vector x [18], i.e., $U^*(k) = f(x(k))$, with $U^* = \left[u_k^*, \dots, u_{k+N-1}^*\right]$. In [18] it is shown that the state-feedback solution f to problem (2) is a piecewise linear function of the states. That is, the states space is partitioned into polyhedra and in each polyhedron the optimal controller is a linear function of the states. Hence, at each time step, the online solution of (2) is reduced to the evaluation of f. This consists of searching for the polyhedron containing $x(k)$ and computing the corresponding linear function in order to obtain U^*. Once U^* is obtained, the first input signal u_k^* is applied, and the procedure is repeated over a shifted horizon. Note that f is time-invariant and it is computed off-line only once.

C. Saccade Controller

When the smooth pursuit controller fails the saccade controller is used. In this case we want to complete the task as fast as possible, hence the saccadic motion. For this reason, the saccade controller, in general, requires a more accurate model of the target motion in opposite to the smooth pursuit controller, which is a robust controller. The need of a more accurate model is translated in the assumption that the disturbance acting on the target is known and constant.

We design a saccade controller by computing the state feedback closed-loop solution to the following constrained finite-time optimal control problem [14]:

$$\begin{aligned}
\min_{N, u_k, \dots, u_{k+N-1}} \quad & \sum_{j=1}^{N} \left\|Qx_{k+j|k}\right\|_p + \|Ru_{k+j-1}\|_p \\
s.t. \quad & x_{k+1|k} = Ax_{k|k} + Bu_k + E\bar{v} \\
& x_{k+j|k} \in \mathbb{X} \quad j = 1, \dots, N \\
& u_{k+j|k} \in \mathbb{U} \quad j = 0, \dots, N-1 \\
& x_{t+N|t} \in T_{set}
\end{aligned} \tag{3}$$

where the disturbance is assumed constant over the horizon and with a terminal state constraints T_{set}. The choice of $\bar{v}$ is discussed in Section II-D. For a fixed N, the off-line solution is piecewise affine [11], i.e, the set of feasible states (states that can be brought to T_{set} in N steps) is partitioned into polyhedra and in each polyhedron the optimal control is a linear function of the states. In order to obtain the state-feedback solution to problem (3), we solve (3) off-line for all horizons $N \leq N_{max}$ for a given N_{max}. This procedure yields N_{max} controllers, one for each horizon:

$$u^*(k)_N = f_N(x(k)), \quad N = 1, \dots, N_{max}.$$

These controllers will be referred to as "explicit controllers". The feasible area of the explicit controllers $f_N(x(k))$ will increase with N since as the horizon is getting longer, state vectors which are farer from the terminal set can be driven into the terminal set.

We implement a feedback controller that brings x to the terminal set T_{set} in minimum time by using the explicit controllers $f_N(x(k))$ in a combined way. At each time step, for a given current state $x(k)$, we look for a feasible controller $f_N(x(k))$ with the smallest horizon N. Once this has been found we implement the corresponding optimal control policy. It should be noted that checking whether or not the state vector is inside a feasible region of a controller is simple: the feasible region of a linear MPC controller is a polyhedron.

We remark that in the proposed scheme, the systems state reaches T_{set} in minimum-time, under no model mismatch for all the cases where $x(k)$ is inside the feasible area of one of the explicit controllers. There is a trade off between the real-time computational complexity and the choice of N_{max}. The use of a long horizon N_{max} allows a wider tracking capability (due to the increase of the feasible area). On the other hand a long prediction horizon requires higher real-time computational capabilities, since such controller has a large number of polyhedral regions. In our approach N_{max} is chosen to be the largest possible given the computational limits of our system. If $x(k)$ is outside the feasible area of

all the explicit controllers, we implement a different saccade controller (3) without terminal set and tuned with a high weight on the position error compared to the other weights. This type of controller, tries to minimize the position errors as fast as it can. When the errors becomes smaller the explicit controllers $f_N(x(k))$ become feasible and are used until the state vector enters the terminal set.

As an alternative solution to reduce the complexity of the saccade controller (3) the method in [19] could be applied. In [19] the authors propose a minimum-time low complexity controller. While this approach provides a wider tracking area with a lower controller complexity, the overall performance degrades compared to our approach.

In summary, at each time step, the proposed strategy for the saccade controller consist in implementing the explicit controller with minimum N. If no explicit controller is feasible, then a saccade controller with no terminal set is applied.

D. Switching between Smooth Pursuit and Saccade

Consider the case where the smooth pursuit controller is active and it becomes infeasible (i.e, the target exits the tracking window). Since the smooth controller is robust to disturbances on the target motion, then an acceleration larger than modeled is acting on the target. In order to apply the saccade controller we estimate the acceleration using the last and current states. With this estimation as input to the target model, we implement the saccade controller as described in the previous section. During the operation of the saccade controller, no image processing is performed, and the control is based on the estimated states of the target. This is due to the fact that it is hard to find the target in the image when the relative speed between the camera and the target is high. This also preserve the ballistic nature of saccades seen in primates.

When the saccade is completed, based on the estimated movement of the target, the system starts to look for the target in the center of the image. If the estimates were correct, the target will be found. If the target is not found, the system switches back to EVS mode and starts to scan the expected area of the target.

III. Experimental Results

Inspired by the arrangement of the chameleon visual system, the robotic head used for experiments is composed of two cameras installed each on top of a pan-tilt units. The two cameras can be moved independently covering large sections of the environment thanks to the wide range of motion of the pan-tilt units. At the high level, an algorithm that scans the environment runs continuously. This algorithm is a heuristic algorithm based on the principles that govern global scanning in the chameleon [6], [7].

The system is equipped with a standard PC. In the PC, the image processing module is responsible to find the target in the images. The smooth pursuit controller and the saccade controller are also hosted on the PC: they receive information from the high-level scanning algorithm during scan operation and directly from the image processing and target location module during target tracking operation. They send references to the pan-tilt units controllers which are located outside the PC and are responsible to perform the movements. The head is composed of two pan-tilt units model PTU-D46-17.5 of DirectedPerception. Two flea cameras of PointGrey are mounted on top of the pan-tilt units and are working with a frame rate of 30Hz. The cameras are connected to a PC computer using a firewire link and the controllers of the pan-tilt units are connected to the computer using a serial RS232 cable.

The smooth pursuit and saccade controllers were first simulated in matlab and then experimentally validated on the robotic head. A short discussion on the system delays can be found in [6], [7]. Next we present two experimental tracking tasks using one camera. The results for tracking a target (black circle) moving along a sinusoidal path with constant amplitude and different frequencies can be seen in Figure 4. When the frequency is low, target acceleration is low and the smooth pursuit controller performs a good tracking. When the frequency is high, the saccade controller is activated due to the high acceleration during the direction change, and drives the target back to the smooth-pursuit tracking window (Figure 4(d)).

Figure 5 depicts experimental results when the target is moving with a constant acceleration. Each experiment begins with an initialization phase that brings the target to a base speed. The base speed is maintained for 0.25 seconds. This is represented by the phase of zero acceleration in the upper plots of Figure 5. After this phase, the test acceleration is applied until the target exits the experiment area. As the acceleration increases, the rate of saccades needed to track the target increases as well. With larger accelerations than the ones reported in Figure 5, after the first saccade, the system could no longer find the target.

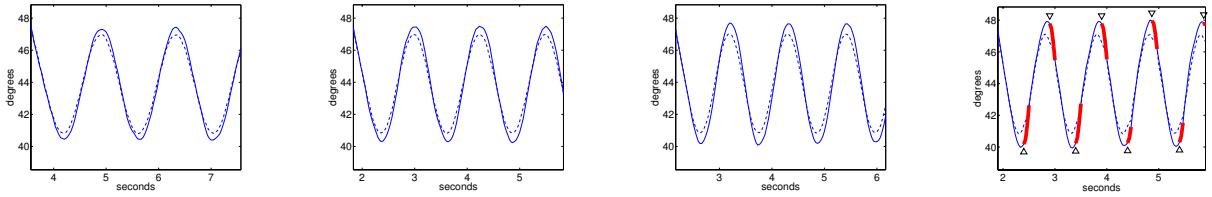
IV. Conclusions

In this paper we have presented a novel method for autonomous target tracking by means of independent pan-tilt cameras which mimic the chameleon visual system. We have designed and implemented minimum time and robust MPC controllers. The MPC technique has several advantages: (i) it is model based, (ii) it allows to include the physical constraints of the system and the target in the control design, (iii) it is robust to abrupt and bounded changes in target directions. Thanks to most recent developments on MPC theory we have computed and implemented the piecewise affine state feedback solution to the MPC problem. The implementation is real-time capable and experimental results have shown excellent performance.

References

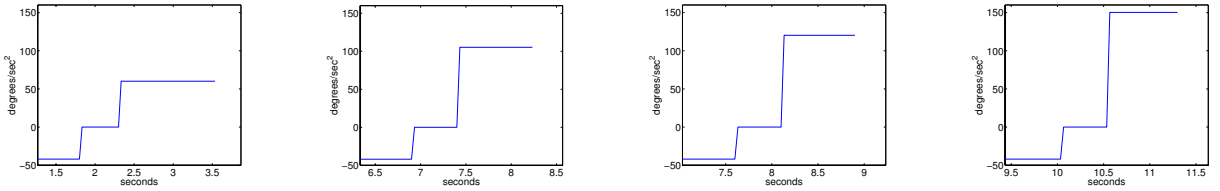
[1] A. Bernardino and J. Santos-Victor, "Binocular visual tracking: Integration of perception and control," *IEEE Transactions on Robotics and Automation*, vol. 15(6), pp. 1937–1958, 1999.

[2] S. Vijayakumar, J. Conradt, T. Shibata, and S. Schaal, "Overt visual attention for a humanoid robot," in *Proc. of IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, vol. 4, October 2001, pp. 2332–2337.

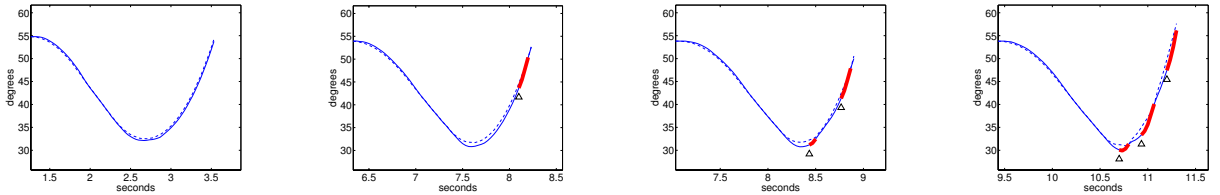


(a) $\omega = 0.7$Hz (b) $\omega = 0.8$Hz (c) $\omega = 0.9$Hz (d) $\omega = 1$Hz

Fig. 4. Experimental results for sinusoidal signal - Each plot shows the target and camera position (in degrees) vs. time (in seconds) for different frequency ω of the signal. The target is represented as a dashed line and the camera in solid line. Saccades are indicated with a thicker line. The start of a saccade is marked with a triangle directly below or above the line. In sub-figure 4(d) the system uses the saccade controller to bring back the target into the tracking window.



(a) Final acceleration: $v = 60°/sec^2$ (b) Final acceleration: $v = 105°/sec^2$ (c) Final acceleration: $v = 120°/sec^2$ (d) Final acceleration: $v = 150°/sec^2$



(e) Final acceleration: $v = 60°/sec^2$ (f) Final acceleration: $v = 105°/sec^2$ (g) Final acceleration: $v = 120°/sec^2$ (h) Final acceleration: $v = 150°/sec^2$

Fig. 5. Experimental results for constant acceleration - plots (a) to (d) show the acceleration profile of the target (in $degrees/sec^2$) vs. time (in seconds). Plots (e) to (h) show the position of the target and the camera (in degrees) vs. time, for the corresponding acceleration profile in the upper plots. The target is represented as a dashed line and the camera in a solid line. Saccades are marked as in Figure 4. The constant acceleration is applied after the target is brought to a base speed, which is maintained for 0.25 seconds. The value of the constant acceleration v is provided in the plots labels. As the acceleration increases, more saccades are needed to track of the target.

[3] M. Asada, T. Tanaka, and K. Hosoda, "Adaptive binocular visual servoing for independently moving target tracking," in *Proc. of IEEE Intl. Conf. on Robotics and Automation (ICRA'00)*, vol. 3, 2000, pp. 2076–2081.

[4] J. D. Pettigrew, S. P. Collin, and M. Ott, "Convergence of specialised behaviour, eye movements and visual optics in the sandlance (teleostei) and the chameleon (reptilia)," *Current Biology*, vol. 9, no. 8, pp. 421–424, April 1999.

[5] M. F. Land, "Motion and vision: why animals move their eyes," *J Comp Physiol A*, vol. 185, no. 4, pp. 341–352, October 1999.

[6] O. Avni, F. Borrelli, G. Katzir, E. Rivlin, and H. Rotstein, "Scanning the environment with two independent cameras - biologically motivated approach," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006.

[7] ——, "Scanning the environment with two independent cameras - biologically motivated approach," Università del Sannio, Benevento, Italy, Tech. Rep. TRxxx, Aug. 2006. [Online]. Available: http://www.grace.ing.unisannio.it/publication/134

[8] E. Rivlin and H. Rotstein, "Control of a camera for active vision: Foveal vision, smooth tracking and saccade," *Int. J. Comput. Vision*, vol. 39, no. 2, pp. 81–96, 2000.

[9] O. Sutherland, H. Truong, S. Rougeaux, and A. Zelinsky, "Advancing active vision systems by improved design and control," in *ISER '00: Experimental Robotics VII*. London, UK: Springer-Verlag, 2001, pp. 71–80.

[10] D. Mayne, J. Rawlings, C. Rao, and P. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, June 2000.

[11] A. Bemporad, M. Morari, V. Dua, and E. Pistikopoulos, "The Explicit Linear Quadratic Regulator for Constrained Systems," *Automatica*, vol. 38, no. 1, pp. 3–20, Jan. 2002.

[12] F. Borrelli, M. Baotic, A. Bemporad, and M. Morari, "Dynamic programming for constrained optimal control of discrete-time hybrid systems," *Automatica*, vol. 41, pp. 1709–1721, 2005.

[13] M. Kvasnica, P. Grieder, and M. Baotić, *Multi-Parametric Toolbox (MPT)*, http://control.ee.ethz.ch/mpt/, 2004.

[14] F. Borrelli, *Constrained Optimal Control of Linear & Hybrid Systems*. Springer Verlag, 2003, vol. 290.

[15] M. Ott and F. Schaeffel, "A negatively powered lens in the chameleon," *Nature*, vol. 373, pp. 692–694, February 1995.

[16] M. Ott, F. Schaeffel, and W. Kirmse, "Binocular vision and accommodation in prey-catching chameleons," *Journal of Comparative Physiology A: Sensory, Neural, and Behavioral Physiology*, vol. 182, no. 3, pp. 319–330, February 1998.

[17] M. Ott, "Chameleons have independent eye movements but synchronise both eyes during saccadic prey tracking," *Experimental Brain Research*, vol. 139, no. 2, pp. 173–179, July 2001.

[18] A. Bemporad, F. Borrelli, and M. Morari, "Min-max Control of Constrained Uncertain Discrete-Time Linear Systems," *IEEE Transactions on Automatic Control*, vol. 48, no. 9, pp. 1600–1606, Sept. 2003.

[19] P. Grieder, P. Parrilo, and M. Morari, "Robust Receding Horizon Control - Analysis & Synthesis," in *IEEE Conference on Decision and Control*, Maui, Hawaii, Dec. 2003, pp. 941–946.