Model Predictive Control to Improve Visual Control of Motion: Applications in Active Tracking of Moving Targets

João P. Barreto, Jorge Batista, Helder Araújo
Institute of Systems and Robotics
Dept. of Electrical Engineering
University of Coimbra
3030 Coimbra - Portugal
jpbar@isr.uc.pt, batista@isr.uc.pt, helder@isr.uc.pt

Abstract

This paper deals with active tracking of 3D moving targets. Visual tracking is presented as a regulation control problem. The performance and robustness in visual control of motion depends both on the vision algorithms and the control structure. Delays and system latencies substantially affect the performance of visually guided systems. In this paper we discuss ways to cope with delays while improving system performance. Model predictive control strategies are proposed to compensate for the mechanical latency in visual control of motion.

1. Introduction

Visual control of motion is a major issue in active vision that involves complex topics of both visual processing and control [1]. This work discusses the problem of tracking moving targets using visual information to control camera motion. An architecture to achieve this goal is presented.

Visually guided systems are dynamic systems whose actions are derived directly from image information. Several strategies to extract visual information for motion control have been proposed [2, 3]. The visual processing must be fast, accurate and robust to achieve high performance behaviors. Mounting the camera on an active platform arises additional difficulties due to the self-induced image motion (egomotion). These problems are discussed and solutions are presented. Kalman filtering is used to estimate the target 3D parameters of motion and limit the effects of measurement errors in the image, allowing smooth tracking behaviors.

Delays in both feedforward and feedback paths of a dynamic system affect substantially the overall performance. This subject is exhaustively discussed in [4]. The latency

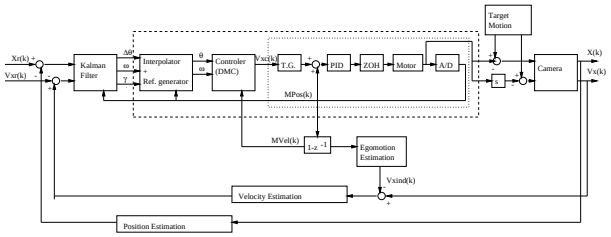

Figure 1. Smooth pursuit block diagram.

introduced by visual feedback is one of the reasons that make vision-based control so difficult. This paper focus on strategies to cope with delays in visual control of motion. The performance of our tracking system is increased by compensating the visual processing delays. Interpolation assuming a constant acceleration model of motion in 3D space is used. The plant to be controlled (in our case a robot head) also presents delays that affect the overall performance. Model predictive control techniques are proposed to cope with process delay in vision-based control. Plant models are obtained with standard system identification techniques and a dynamic matrix controller (DMC) is used in our tracking application. The performance of the DMC controller is discussed and compared with other possible control strategies.

2. System Architecture

In a monocular tracking application the camera has two independent rotational degrees of freedom: pan and tilt. The control of each of these degrees of freedom can be modeled by the schematic of Fig. 1. The system inputs (references) are the desired target position and velocity in the image, and the outputs are the actual target position and

velocity in the image. This is typically a regulation control problem whose goal is to keep the moving target in a certain position in the image (usually its center).

In a real-time tracking systems we can identify three distinct concurrent processes: the vision processing of the images, the servo control of camera platform and the smooth pursuit controller (see 1). The visual feedback loop typically runs at 25Hz. A high gain, high sample rate local servocontroller is needed to ensure that close control over platform position and velocity is maintained with minimum error. Several dedicated servo control modules are commercially available. In our system, platform motion is generated by DC motors equipped with optical encoders for position feedback. Each axis is controlled by an independent module that implements a closed loop with a digital PID filter running at 1KHz. Each servo loop can be commanded in velocity by adding a profile generator that integrates the velocities sent by the user process. Communication is synchronous at a frequency of 166Hz. This means that user process can only send commands to the servo module and read the encoders in every 6ms time intervals. The smooth pursuit controller makes the interface between the high level visual loop and the low level servocontroller. It receives the result of visual processing and sends velocity commands to the servo loop. This middle level controller must run at the maximum communication rate (in our case 166Hz) to optimize global system performance.

3 The Visual Control Loop. Visual Processing Delays

Target motion acts as a perturbation that has to be compensated for. To implement high performance tracking certain issues, such as robustness to sudden changes of target trajectory and velocity, can not be neglected. Evaluation of both vision and control algorithms within a common framework is needed for the optimization of the global system performance [5]. This framework has been established in previous work [6]. To study and characterize the system regulation/control performance usual control test signals (step, ramp, parabola and sinusoid) must be applied.

Visual processing latency compromises the overall system performance. Computation time in extracting information from images must be minimized. A trade-off between efficiency, robustness and accuracy must be made when selecting the visual processing algorithms. Simultaneous position and velocity information are fundamental to achieve high performance smooth tracking behaviors. Thus both target position and velocity in image must be measured.

Image motion depends both on target motion and camera motion (egomotion). For visual control tasks we are only interested in the motion induced by targets, thus egomotion must be compensated for. Considering that the camera only performs pure rotations and that there is no motion in the scene, two images are related by an homography. The homography is easily computed knowing the camera rotation measured using the motor encoders. Considering two consecutive frames, the difference image obtained after egomotion compensation contains the points where motion occurred. Position is estimated as the average location of the set of points with non-zero optical flow and non-zero brightness partial derivatives, with respect to X and Y, in the most recently grabbed image. It is assumed that all moving pixels in image have the same velocity. The velocity vector is estimated considering the flow constraint and applying a least-squares minimization. Gaussian pyramids are used to increase the range of image velocities that can be correctly estimated [6]. A Kalman filter is used to estimate the target angular parameters of motion (error in position $\Delta\theta$, velocity ω and acceleration γ) assuming a constant acceleration model between frames. This assumption is acceptable for frame acquisition rates of 25Hz and higher. The inputs to the Kalman (innovation) are the position and velocity measurements in the image and the output the state vector $\mathbf{x}$. The Kalman filter tuning has been performed with the help of our evaluation tools. Each grabbed image is processed (in a standard PC) and the parameters of motion of the target are available at the Kalman filter output 6ms after image acquisition. This is the visual latency. A structure with the estimated parameters, the image acquisition time and motor position at the acquisition time instant is sent by the high-level process (running at 25Hz) to the smooth pursuit controller (running at 166Hz). In our first control strategy the velocity command sent to the low-level loop is the sum of

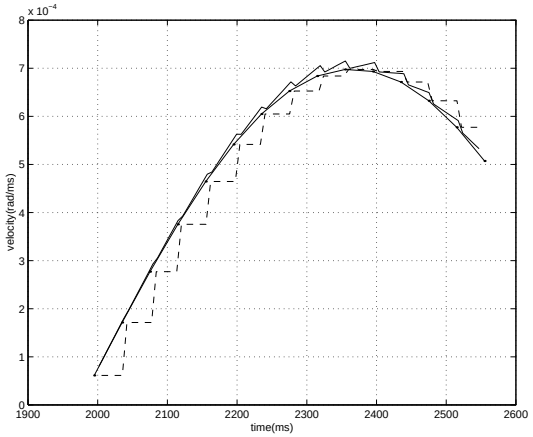

Figure 2. Target angular velocity (.-). Velocity command sent to actuator: initial implementation (–), using interpolation for visual delay compensation(-)

target velocity and tracking error in position multiplied by a gain K_p. The position component is fundamental to keep the regulation error small and to reduce the effects of occasional errors in velocity prediction. However the target motion information is sent to actuators with a delay of 6ms and kept constant until new visual information is received (see 2). This really compromises system performance. Interpolation assuming a constant acceleration model is used to compensate for the visual delay. Fig.2 compares the velocity command sent to the low-level loop with and without interpolation. Notice ripple at the top of the sinusoid, due to the highly non-linear variation of target velocity that is not described by the constant acceleration model used for interpolation.

4. Actuator/Plant Delay

Standard system identification techniques can be used to obtain the transfer function of the low-level control loop. The input considered is the velocity command sent to the profile generator (V_{xc}) and the output is the motor velocity (M_{vel}). Notice that the low-level loop runs at 1KHz and its output is being subsampled at 166Hz. Thus, the achievable model will not describe some high frequency behaviors. For our implementation the transfer function has a deadbeat of 2 sampling periods. It means that actuator/plant delay is nearly 12ms. This section discusses the use of model predictive controllers to cope with this delay.

$$J = \sum_{i=N_1}^{N_2} (y(n+i|n) - w(n+i))^2 + \sum_{j=N_1}^{N_2} \lambda \Delta u(n+j-1)^2 \quad (1)$$

There is a wide variety of MPC algorithms, but they always have three elements in common: a prediction model, an objective function and a minimization process to obtain the control law. The prediction model is used to estimate the system output $y(n+k|n)$ at future time instants knowing previous inputs and outputs. The general aim is to make future system outputs to converge for a desired reference $w(n)$. For that an objective function J is established. The general expression for such a function is given by equation 1. N_1 and N_2 bound the cost horizon, N_u is the control horizon, $u(n)$ is the control signal, $\Delta u(n)$ is the control increment ($\Delta u(n) = u(n) - u(n-1)$) and λ is relative weight used to achieve a more or less smooth control. In order to obtain present and future values of control law $u(n)$ the functional J is minimized.

The cost horizon is the future time interval where it is desirable for the output to follow the reference. Our process has a dead time of 2, thus we are going to consider $N_1 = 2$ (the output can not be forced before that). Assuming a frame rate of 25Hz, the middle level controller sends at most 7 velocity commands to the low-level loop without new visual information. Thus we are going to consider $N_2 = 8$.

Consider the step response $g(n)$ of a stable linear process without integrators. If $g(n) = 1$ for $n > N$ the system is completely described by the N first instants of $g(n)$. This is the cornerstone for a simple, robust and intuitive model predictive controller: the dynamic matrix control algorithm.

$$\boldsymbol{\Delta u} = (\boldsymbol{G}\boldsymbol{G}^{\boldsymbol{t}} + \lambda \boldsymbol{I})^{-1}\boldsymbol{G}^{\boldsymbol{t}}(\boldsymbol{w} - \boldsymbol{f}) \quad (2)$$

DMC uses the N first instants from the step response to predict the system output (in our case $N = 7$). It assumes a constant disturbance along the cost horizon. The disturbance is given by the difference between the actual system output and the predicted output ($d(n) = y(n) - y(n|n)$). The goal of our controller is to drive the output as close as possible to the reference in the least-squares sense. The control action for that is computed by equation 2. $\mathbf{G}$ is the dynamic matrix of the system, $\boldsymbol{\Delta}\mathbf{u}$ is the control vector and $\mathbf{w}$ is the reference vector. $\mathbf{f}$ is called the free response vector because it does not depend on the future control actions. λ is the penalty for the control effort, by increasing this value the system becomes less responsive and smoother. Notice that only the first element of $\boldsymbol{\Delta}\mathbf{u}$ is really sent to the motor. The vector is computed at each iteration to increase the robustness of the control to disturbances in the model. For more details on DMC controllers see [7].

Interpolation can be used, not only to compensate for the visual processing delay, but also to estimate target parameters of motion for future time instants. Visual information at the Kalman filter output is used to compute current and future target angular position and velocity assuming a constant acceleration model of motion. The goal of the DMC controller is to force the motor to have the same motion as the target in a near future. Considering that $\mathbf{w}$ is the desired motor velocity, it would be reasonable to use the predicted target velocity as reference for the DMC controller. The result of this control strategy can be observed in Fig.3(M)(D). Notice that by using only velocity information the system is not able to compensate for errors in position and the target is lost after a certain period. Despite that the motor successfully reaches target velocity.

$$\Delta_p = \frac{1}{6}\left(\frac{E_p}{T} - MV_o\right) - \frac{\Delta_v}{3}\left(\frac{M^2}{4} + 2\right) \quad (3)$$

Whenever a new image is grabbed, visual processing is used to compute target velocity and tracking position error. Perfect tracking is achieved if, at the next frame time instant, the system compensates for the error in position and moves at the estimated velocity. This is the goal considered to establish the reference $\mathbf{w}$ whose profile is depicted in Fig. 3(U). Consider P_o and V_o are the current motor position and velocity and $P_t(i)$ and $V_t(i)$ are the target position and

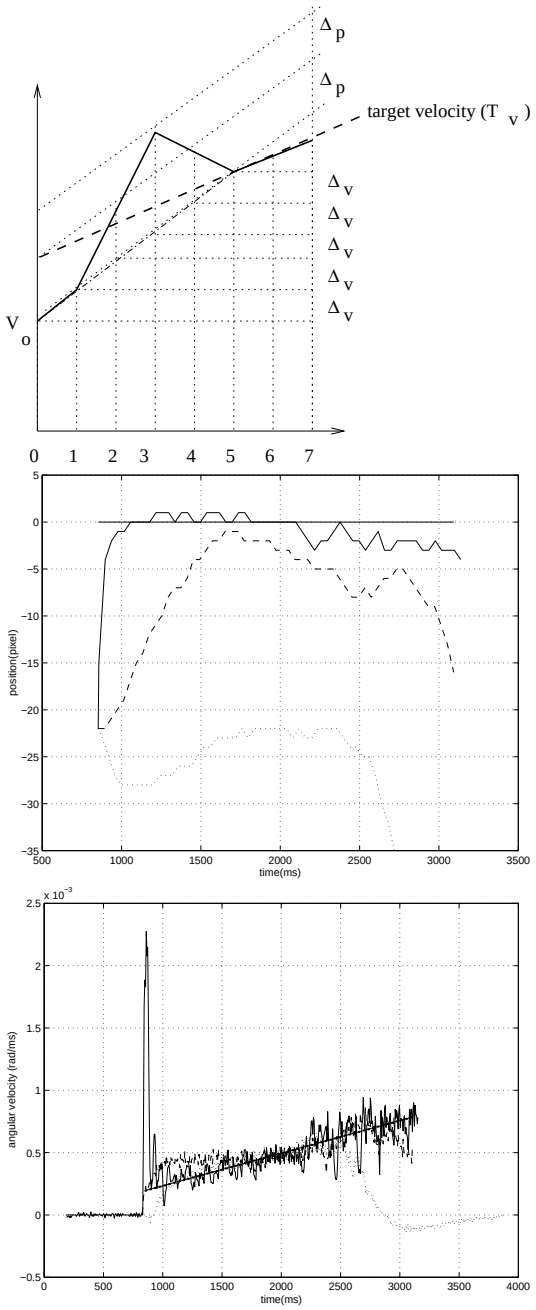

Figure 3. Top: Deriving the velocity reference trajectory: (–) is the target estimated velocity , and (-) is the reference velocity. Middle and Down: Tracking a target with a parabolic trajectory of motion: initial position 5deg, initial velocity 10deg/s, acceleration 15deg/s^2. Middle: Regulation performance. Target position in the image. Down: Regulation in angular velocity. Target (.-), First controller(–), DMC velocity controller (:), DMC velocity +position controller(-)

velocity at instant i. Then $\Delta_v = (V_t(M) - V_o)/M$ and Δ_p is computed by equation 3 where $E_p = P_t(M) - P_o$. M is the instant of convergence, making $M = 5$ motor velocity converge to target velocity in 5 samples (30ms). In this time interval the motor accelerates and then slightly decelerates to compensate for the position error. The velocity reference $\mathbf{w}$ is computed for each control iteration.

The increase in performance introduced by the DMC controller can be observed in Fig3. The error in position is immediately compensated and the target is kept in the center of the image along its parabolic motion. In the velocity regulation figure notice the initial peak that compensates for the position error. At the end the velocity regulation performance decreases for the three controllers. This is due to visual processing limitations. The velocity estimation algorithm is not able to measure velocities above a certain threshold. This has been discussed in previous work and Gaussian pyramids have been introduced to cope with this problem.

References

[1] S. Hutchinson, G. Hager, and P.I. Corke. A tutorial on visual servo control. *IEEE Trans. on Robotics and Automation*, 12(5):651–670, October 1996.

[2] K. Pahlavan, T. Uhlin, and J. Ekhlund. Integtaring primary ocular processes. *Proc. 2nd European Conf. on Computer Vision*, pages 526–541, 1992.

[3] E. D. Dickmanns, B. Mysliwetz, and T. Christians. An integrated spatio-temporal approach to automatic visual guidance of autonomous vehicles. *IEEE Trans. on Systems, Man and Cybernetics*, 20(6):1273–1284, November/December 1990.

[4] P. M. Sharkey and D. W. Murray. Delay versus performance of visually guided systems. *IEE Proc.-Control Theory Appl.*, 143(5):436–447, September 1996.

[5] P.I. Corke. Visual control of robot manipulators–a review. In K. Hashimoto, editor, *Visual Servoing*. World Scientific, New York, 1993.

[6] João P. Barreto, Paulo Peixoto, Jorge Batista, and Helder Araujo. Evaluation of the robustness of visual behaviors through performance characterization. In Markus Vincze and Gregory D. Hager, editors, *Robust Vision for Vision-Based Control of Motion*. IEEE Press, 1999.

[7] E. F. Camacho and C. Bordons. *Model Predictive Control*. Springer-Verlag, 1999.