A Biologically Inspired Controller For Fast Eye Movements

Martin Lesmana and Dinesh K. Pai

Abstract—We describe and test a non-linear control algorithm inspired by the behavior of motor neurons in humans and other animals during extremely fast saccadic eye movements. The algorithm is implemented on a robotic eye, which includes a stiff camera cable, similar to the optic nerve, which adds a complicated non-linear stiffness to the plant. For high speed movement, our “pulse-step” controller operates open-loop using an internal model of the eye plant learned from past measurements. We show that the controller approaches the performance seen in the human eye, producing fast movements with little overshoot. Interestingly, the controller reproduces the main sequence relationship observed in animal eye movements.

I. INTRODUCTION

Eye movements are some of the fastest and most accurate movements made by animals and provide a tantalizing window into biological motor control algorithms [1]. We are motivated by two intertwined questions: (1) Can biological theories of motor control proposed in neuroscience be effective for controlling real robots? (2) Can we gain insights into biological motor control by building physical robots [2], [3], rather than simulations of animal eyes? In this paper we investigate both these questions with a simple, 1-DOF, robotic eye designed to make extremely fast movements, called saccades, with a performance similar to that of the human eye. We emulate the dynamics of a human eye using this robot, and design a biologically inspired nonlinear “pulse-step” controller to control saccadic eye movements. During the high speed movement, the controller operates open-loop using an internal model of the eye plant learned from past measurements.

Animals do not scan their environment by continuous eye movement but by alternating fixation and saccades. Saccades are the fastest movements made by animal eyes; a saccade of less than 10 degrees can be completed in less than 50 ms [4]. One reason is that during movement, motion blur significantly degrades vision; therefore these movements must be completed as quickly as possible without losing too much positioning accuracy. There is a popular misconception that saccades are necessary only for animals with foveas (i.e., which have high resolution in a small angle of the visual field, about 2 degrees in diameter in humans) and that this does not apply to robot cameras. However, even afoveate animals like rabbits need to make saccade-like movements since they have a limited field of view, and need to shift it when nearing the “joint limits” (this is called quick phase nystagmus) [5].

It is difficult to use feedback control for these high speed movements, in both animals and robots. Neural signal processing of sensory data is relatively slow in animals; there is significant evidence that saccades are performed in an open-loop fashion without visual or proprioceptive feedback [6], [1]. Signal processing in robots can be faster but traditional feedback control requires high gains and accurate estimation of derivatives to achieve the speeds seen in animal eyes, but such a controller can easily become unstable. Instead, there is strong experimental evidence that both humans [7] and animals [8] use a non-linear pulse-step controller (discussed in section II).

Contributions

In this paper we implement a controller that is inspired by the biological controller used to direct fast and accurate eye movements in humans. This controller is non-linear and allows for fast movements with no overshoot as it does not follow a typical asymptotic approach to the final position. A classical linear controller such as the PID controller would require very high gains to achieve these fast movements. The plant model used in designing a PID controller is also often assumed to be linear or a linear approximation of it. However, for a system such as the one used in our experiments (see section IV-A) a linear assumption is unsuitable due to the strong non-linearity introduced by the cable. The controller described in this paper automatically learns a general non-linear plant model by itself and skips the tuning process to create a control strategy that is best suited for the plant.

The remainder of the paper is organized as follows. Section II gives the high level overview of the controller that is developed. Section III describes the technique used to estimate a non-linear model for the plant. Section IV provides the details of the controller implementation and the setup used to test it. The results from the simulation and experiment are discussed in section V. We conclude with a discussion of the present and future work in section VI.

Related Work

Motion control in industrial robots are often achieved through the use of a proportional-derivative (PD) feedback controller. This can be further augmented by an integral term to improve steady state performance and by a feedforward term to improve its dynamic performance [9]. The design of these different controller components require some knowledge of the plant being controlled. Furthermore, an online learning system such as described in [10] can be used to improve the controller by a learning process that
can capture the inaccuracy of the plant model used. Other common control techniques used for robotic manipulators include adaptive control [11], [12], sliding mode control [13], and computed torque control [14].

There has been much work on various robotic eye/gaze platforms by the active vision community. They include biologically inspired [15], [16], [17], [18], [19], [20], and also non-biologically inspired designs [21]. Specifically, the mapping of target in the visual field to saccade target position has also been investigated [22], [23]. The actual generation of saccadic motion itself has been achieved by using a priori knowledge of the plant geometry and dynamic [24], tracking a spline trajectory using an inverse dynamic controller [25], or tracking a velocity profile [26]. An implementation of catch-up saccades during smooth pursuit (tracking) eye motion has been described in [27].

Although some of these works involve the generation of saccadic-type motion, we are not aware of any work on controlling rapid (saccadic) eye movement that exhibits the pulse-step behavior seen in the biological counterpart for a robotic platform. It is worth noting that torque pulse, similar to the pulse-step, has been used to generate fast motion for robotic platform [28], [29] by learning the mapping from visual input to pulse parameters.

II. A NON-LINEAR CONTROLLER FOR RAPID MOVEMENTS

Our controller is based on the “pulse-step” controller proposed in neuroscience for control of saccadic eye movements. The control signal sent to oculomotor neurons to generate a saccade is composed of two main components as shown in Fig. 1a. It starts with a very high constant signal called the pulse followed by a lower constant signal referred to as the step. While the step signal relates to the steady state holding force to keep the eye at the target position, the magnitude of the pulse component is roughly same regardless of the target location or saccade amplitude [1]. Therefore, the amplitude of the saccade is modulated primarily by its duration. This model is a simplification since there is usually a “slide” region between the pulse and the step, and other variations, but it is a good first approximation.

From a control point of view, this is a switching controller which switches from a fast movement phase to a fixation phase based on the distance to target. The controller action can be physically interpreted as trying to activate the muscles to move the eye as fast as possible to the target position before applying the appropriate force level to hold it at its target. Although this controller is similar to a bang-bang controller, unlike a bang-bang controller, its output does not only switch between the boundaries of the feasible control signal \( u(t) = M_+ \) or \( u(t) = M_- \) (where \( M_+ \) and \( M_- \) are the bounds of the feasible control signal) [30]. Instead, the pulse-step controller uses the entire range of feasible control signal.

The pulse-step controller uses an internal forward model simulation to determine the proper instance to transition from the pulse to the step. The forward model is a learned (estimated) non-linear model of the system plant. We discuss the learning process in section III. We simulate the actual plant trajectory by exciting the forward model with a copy of the input to the plant and integrate it forward in time. In neuroscience, the copy of the input signal sent to the muscle is referred to as the efference copy.

A system level overview of the pulse-step controller as described is compared to the classical PD controller in Fig. 1b. We can see that while the PD controller feeds the output signal back into the controller (hence the feedback), we also see a similar feedback line in the pulse-step controller. In the pulse-step controller however, the signal that is fed back is the control signal rather than the plant output (sensor measurement). The forward model then transforms the control signal into an estimate of the system state which also contains the output. This is referred to as an internal feedback [31] and it allows the controller to have knowledge of the output without having to actually measure it.

Once the internal model reaches the target position, the controller generates a step signal that is required to hold the system at that position. The appropriate magnitude of the step signal can be determined directly from the inverse model of the learned plant by evaluating the force at the target location in steady state condition.

III. NON-LINEAR PLANT MODEL LEARNING

In this section we describe the method used to learn a non-linear plant model.

We first assume a very general second order plant:

\[
u = J\ddot{\theta} + F(\theta, \dot{\theta}).
\]

This system contains a linear expression for the highest order (inertial) term \( J \) and a non-linear force term \( F(\theta, \dot{\theta}) \); \( u \) is the applied torque and \( \theta, \dot{\theta}, \text{ and } \ddot{\theta} \) are the position, velocity, and acceleration respectively.

The last term in (1) is a non-linear function of the state space \( (\theta, \dot{\theta}) \). We parameterize it by a linear combination of \( n \) radial basis functions (RBF) \( (B_k(\theta, \dot{\theta})) \) placed regularly in the state space. An RBF is defined as a real function that is dependent only on the distance of its input from its center [32]. The choice for the RBF is the non-compactly supported squared Mahalanobis distance for its cheap computational cost (this is important when we are trying to evaluate them real-time).

Thus the model of the plant can be reexpressed as:

\[
u \approx J\ddot{\theta} + \sum_{k=1}^{n} w_k B_k(\theta, \dot{\theta}),
\]

where \( w_k \) is the weight for the \( k'\text{th} \) RBF. Hence, to learn this parameterized system model, we simply need to learn the values of \( J \) and \( w_k \)'s.

Since this RBF is weighted most heavily around its center, it has a useful property of preserving the locality of information. However, this means that each of the RBF is effective only in approximating the non-linear function locally with the locality determined by its covariance. Therefore we need
A networked computer running xPC target (MathWorks Inc.) is connected to the motor amplifier and encoder. It acts as a hard real-time controller with a sample time of 0.5 msec. It implements a virtual spring-damper system for the camera platform by means of a PD controller. The spring and damper coefficients are chosen so as to match the dominant time constant of the human eyeball (0.299 seconds) as obtained experimentally by Robinson [33]. Consequently, the overall plant that we are controlling is overdamped, similar to the behavior exhibited by the mechanics of the human eye.

B. Pulse-Step Controller

The real-time controller also executes the pulse-step controller algorithm, independent of the virtual spring-damper. The target locations are specified to the pulse-step controller through an external command sent over the network. The overdamped characteristic of the plant allows the transition from pulse to step to be decided in a simple fashion: we can rely on the damping to bring the system to a stop almost instantaneously when the controller transitions from pulse to step (provided the step level chosen corresponds to the correct amount of holding torque at that position). For a more general case, a look-ahead integration can be performed in addition to the current state estimation at each time step. This simulates the system trajectory up to a short interval in the future for the scenario whereby the control signal is switched to the step level. The integration of the forward model is performed numerically using a Störmer-Verlet integrator which provides us with a better accuracy than an Euler integrator. Due to the very fast motion (about 0.2 degrees per millisecond), the size of the timestep for the forward model integration needs to be sufficiently small to minimize error in the steady state estimate. We use a timestep size of 2 milliseconds in our implementation.

An issue encountered with the controller was the larger than expected steady state error caused by the motor friction. For a given step input sent to the plant, the steady state position reached would have an error of approximately 2 degrees due to friction. Therefore the controller cannot rely on the step level to correct any error in the pulse component of the trajectory. Furthermore, as a result of the rapid trajectory motion, the fast higher order dynamics affects the accuracy of the estimated trajectory of the pulse, causing an RMS error of 1.5 degrees. However, a correlation between the saccade amplitude and the steady state error was
observed and nearly all of the steady state errors are caused by undershoots as shown in Fig. 2. This correlation is likely to be caused by the higher order dynamics having a damping effect. This unmodelled damping effect causes the system to lose more energy than the forward model simulation and this error accumulates with increasing pulse duration (saccade amplitude). We compensate for this extra damping by using the above correlation in the controller to modify the saccade targets by amounts corresponding to the amplitude.

V. RESULTS

A. Simulation Results

We first tested our plant model learning system in simulation. A synthetic pulse-step data is generated with a system composed of a linear mass-spring-damper. Following the dominant time constant in Robinson’s human eye mechanic model, we implement a linear spring and damper system on the experimental setup with a spring constant of 0.5 mNm/deg and a damping constant of 0.15 mNm/(deg/s). The inertia is matched to the experimental setup motor load inertia \((2.08 \times 10^{-7} \text{Nm} \cdot \text{s}^2/\text{deg})\). Fig. 3 shows a series of pulse-step trajectory and force generated on this system. Applying the non-linear model learning gives us a very good estimate of the system parameters with a root-mean-square (RMS) residual of \(6.88 \times 10^{-4} \text{Nm}\). The estimated mass has a relative error of 10\% and the learned \(F(\theta, \dot{\theta})\) is approximately linear, as expected from the the linear spring-damper.

We then cross validate the learned plant model with a new set of simulated pulse-steps trajectory. This gives us a good result, as shown in Fig. 4, with a fit of 99.8\%. Fit is defined as:

\[
\text{fit} = 100 \times \frac{1 - ||y - \bar{y}||}{||y - \bar{y}||},
\]

where \(y\) and \(\bar{y}\) are the cross-validation trajectory and its mean, while \(\bar{y}\) is the trajectory generated from the learned model.

B. Experimental Results

The parameters of the virtual spring-damper in our experiment are matched to those used in the simulation. However, the cable introduces a significant amount of parasitic torque as discussed below.

We first learn a non-linear plant model from a set of pulse-steps trajectory. Even though the implemented virtual spring-damper system is linear, we are affected by unmodelled system dynamics that can come from the amplifier, friction and the camera cable. These dynamics cause the overall system to behave in a non-linear fashion even though a DC motor with load may appear to be linear at first. The non-linearity of such a system is usually hidden away by the robustness of the feedback controller for slow dynamics. However our setup requires us to deal with fast system dynamics, nullifying the above feedback controller advantage.

With the implemented pulse-step controller from Section IV-B, we command the system to move to a series of targets. Fig. 5 shows an example of the controller performance. The steady state RMS error (measured prior to the start of each
new commanded target) is $0.99\degree$. However the accuracy of the controller improves with the cable removed, giving us a steady state RMS error of $0.24\degree$. We attribute this improvement to the uncertainty involved with the parasitic torque introduced by the cable.

Our experimental data exhibits a relationship between saccade amplitude and duration, that is similarly observed in saccadic eye motion of animals. This relationship, often referred to as the main sequence [34], is shown in Fig. 6.

VI. DISCUSSION AND FUTURE WORK

A. Discussion

There are many control strategies commonly used for the control of motion. However most of them only work well with a linear system and stability issues tends to arise as the system performance requirement and the associated controller gains increase. They often require knowledge of the plant which can be difficult to obtain or hobbled by parameter inaccuracies and unmodeled dynamics. We have developed a control system strategy learned from observations of fast eye movements in animals. This controller is unhindered by many of the issues and limitations associated with a typical controller such as the one described above. In fact, it is able to apply sharp changes to the control signal which is not possible with a PD controller for example, without setting its gains to be very high. By implementing this controller on a robotic platform, we have also at the same time shown the plausibility of the use of such a controller for fast eye movements in animal.

A challenge that we encountered in our implementation is with the dynamics introduced by the camera cable. The parasitic torque that it introduces have been at least partially modeled by our learning algorithm. Even so, the performance of the system is still affected by the cable. This indicates that the dynamics of the cable has not been captured fully by our model, possibly due to its high order dynamics or dependence on initial condition. Also, friction in the motor was unexpectedly difficult to model with our learning algorithm. We initially expected its effect to show up as a discontinuity in the state space at zero velocity. Nevertheless such a behavior was not observed. We attribute this to the smoothing or smearing effect caused by the limited sensor performance and possibly unmodeled dynamics.

B. Future Work

An unavoidable aspect of real physical systems as opposed to simulations is that their properties change over time. This can happen over a short term as a DC motor heat up or over a longer period of time due to wear and tear for example. Such changes would lead to a discrepancy between the learned and the actual plant model, deteriorating the feed-forward controller performance. The learned plant model can adapt to these changes by applying an online learning process as opposed to a one-off learning at the beginning. A possible online learning scheme is recursive least squares with exponential forgetting. However, care need to be taken with regard to the use of exponential forgetting: the system is poorly excited by piecewise constant input, generating short transient response with long periods of steady state in between. This can lead to instability in the estimated parameters.

Even with a good system model, steady state error will still persist due to noise. The classical approach to deal with steady state error is by adding an integral action to the feedback controller. Without a feedback system, it is impossible to eliminate steady state error completely. Therefore a feedback mechanism can be combined with the developed pulse-step controller to improve the steady state tracking performance. This can be in the form of a corrective step taken after the main motion (akin to a corrective saccade in human), or with the feedback mechanism taking into effect during the saccadic motion.

A current limitation of the pulse-step control signal is the sharp transition between the pulse and the step. A slide region can be introduced between the two to give a smooth transition. This parallels the observed biological neural pattern.

An alternative to obtaining the trajectory data from instrumenting the actuator (an encoder in our setup) is to instead use the existing camera information. Steady state (targetting) error can be obtained by using a feature detector and determining the distance between the centre of the image to the position of the target of interest on the camera. Furthermore, optical flow information during camera motion can possibly be used to provide velocity (and consequently trajectory) information.

VII. ACKNOWLEDGEMENTS

We would like to thank Dr. P.-E. Forssén and B. Dow for their help in the building of the robotic platform. This work is funded in part by the Canada Research Chairs Program, Peter Wall Institute for Advanced Studies, the Human Frontier Science Program, NSERC, CIHR (CRCNS), and NIH.

REFERENCES

Fig. 6. The main sequence relationship between saccade amplitude and duration

(a) Our experimental result with camera cable disconnected
(b) Our experimental result with camera cable connected
(c) Measured human saccades data fitted with a power equation. Reproduced from [4], by permission of Oxford University Press.