**Abstract**

We introduce a vision-driven biomechanical controllers for virtual human characters. Our character detects a target object using edge detection techniques.Then, the character estimates the target position with stereo vision and kalman filter, and reaches for the target with inverse kinematics.

**Contributions:** Xiaolong worked for "Target trajectory estimation with kalman filter" and Masaki worked for "Object detection with canny edge detector", " Inverse kinematics for arm control", "Depth estimation with stereo vision".

## 1 Introduction

More accurate biomechanical modeling should result in more realistic human animation. Given realistic biomechanical models, however, we must confront numerous motor control problems. The kinematics and dynamics are challenging in human character control due to the complexity of the human body structure.

The goal of our project is the integration of visual functionality in complex human models. In particular, we implement an active vision system in biomechanical human models. We use the Canny edge detector [1] to detect objects in the visual field. After estimating object position, the full body model makes an effort to reach the target with an arm. The resulting animations demonstrate realistic motion with human-like control flow initiated by visual information.

The remainder of this report is structured as follows: In Section 2, we explain a method we use in vision. We explain our model in Section 3, and the control system in Section 4. We cover our experiment result in Section 5, followed by conclusion and future work in Section 6.

## 2 Vision

### 2.1 Object Detection

In our application, object detection is performed on a scene rendered with visual information through the virtual eyes. The objective is to provide the ability of visual processing for a virtual human model.

First, all the visual information from the perspective of the eye is passed into an image structure for further processing. To perform object detection, certain features of objects are detected. Edges are one of the important features of an object, and a fundamental part of the feature detection paradigm.

The Canny edge detector [1] is applied on the monochrome version of the image to detect the intensity edges in the image. We connect those edges to obtain the contour of the target object. The contours give minimal yet useful information about the object. A more sophisticated example of contours is the Active Contour models [5], which can be also used for tracking objects in subsequent frames. The next step is to obtain a bounding box around the contours of the detected objects. The bounding box is a major step as it helps us to characterize the object (e.g., width and height) and to obtain the location of the object. We approximate the midpoint of the target object as follows:

$$x_{mid} = x + \frac{width}{2} \tag{1}$$

$$y_{mid} = y + \frac{height}{2} \tag{2}$$

## 2.2 Stereo Vision

We use stereo vision technique to estimate the depth information. We detect the target with canny edge algorithm and estimate the location; however, this information is only on 2D space with single eye. Humans have 2 eyes, and we can estimate the distance to the target with stereo vision technique. Similar triangle theory is used for the calculation. We compute the distance with following equation.

$$z = b \times f / d \tag{3}$$

where z is the depth distance from eyes, b is a distance between two eyes, f is a distance from eye to screen and d is a disparity. The disparity is computed by the following equation.

$$d = (x_{right} - x_{left}) \tag{4}$$

where x represents the displacement of the target projected on each screen.

## 2.3 Visual Estimation of Target Movement Using Kalman Filter

Human vision is remarkably complex and the subject of intense ongoing study. For computer animation, we need a relatively simple model that captures the relevant features of human vision, but by no means all the complexities. Similar to [6], we use a Kalman filter for predicting the state (position and velocity) of ball, using plausible models of noise introduced by visual sensing. It is noted that in addition to the noise generated by the environment as [6], our model uses stereo vision to estimate depth information, which inherently contains noise, see section "xxxx." However we believe this noise generated by stereo vision is controllable and better represents errors introduced by human visual sensing.

Table 1: Parameters for error standard deviations of foveal vision: units for last three rows are identical to their original unit

| | Parameter | Value |
|---|---|---|
| Retinal Position | $\sigma_y, \sigma_z$ | $2.9 \times 10^{-4} rad$ |
| Retinal Velocity | $\sigma_{\dot{y}}, \sigma_{\dot{z}}$ | $0.05\lvert\dot{q}\rvert$ |
| Depth | $\sigma_x$ | $0.03d$ |
| Depth Velocity | $\sigma_{\dot{x}}$ | $0.05\lvert\dot{d}\rvert d$ |

We define a spatial coordinate frame attached to the eye, with it's center at the middle of two eyes in our biomachenical model, X-axis aligned with the visual axis aligned with the visual axis, and Z-axis vertical in a reference position looking straight ahead. For ball catching, we define the perceived position and velocity of the ball as a six dimensional state vector in the eye coordinate frame, $\boldsymbol{x} = \binom{\boldsymbol{p}}{\dot{\boldsymbol{p}}}$

## 2.4 Vision

We approximate the probability of the state of a moving point seen by the eye as a multivariate normal distribution in the eye coordinate frame. The probability of the perceived position $\boldsymbol{p}$ and $\dot{\boldsymbol{p}}$ of the target in the eye-fixed frame is:

$$\begin{bmatrix} \boldsymbol{p} \\ \dot{\boldsymbol{p}} \end{bmatrix} = \boldsymbol{N}(\begin{bmatrix} \bar{\boldsymbol{p}} \\ \dot{\bar{\boldsymbol{p}}} \end{bmatrix}, \boldsymbol{A\Sigma A})$$

$$\boldsymbol{\Sigma} = \frac{1}{\alpha(\theta)} diag([\sigma_x, \sigma_y, \sigma_z, \sigma_{\dot{x}}, \sigma_{\dot{y}}, \sigma_{\dot{z}}]$$

$$\boldsymbol{A} = \begin{bmatrix} e^{[\omega]\theta} & \boldsymbol{O_{3\times3}} \\ \boldsymbol{O_{3\times3}} & e^{[\omega]\theta} \end{bmatrix}$$

where $\bar{\boldsymbol{p}}$ is position of the ball in 3 dimensional space retrieved from stereo vision and $\dot{\bar{\boldsymbol{p}}}$ is the true velocity of the ball, $\boldsymbol{\Sigma}$ is the error covariance matrix in eye frame and $\boldsymbol{A}$ the corresponding transformation matrix from the target to the eye center. The block diagonal elements of $\boldsymbol{A}$ represent the rotation needed for the eye to foveate the target and $[\omega]$ is a skew symmetric matrix of the axis of this rotation. It is noted that the movement of eyeball is not yet implemented in current version of our model.

The noise parameters above are chosen the same as Table 1.

## 2.5 Internal Model and Bayesian State Estimation

Using the vision model described above, a noisy observation of the target is obtained. The state update speed is the same as the global animation environment, $\Delta t = 10ms$

The internal model of the ball's dynamics is represented as

$$\boldsymbol{x_{k+1}} = \boldsymbol{F x_k} + \boldsymbol{b}$$

$$\boldsymbol{F} = \begin{bmatrix} \boldsymbol{I_{3\times3}} & \boldsymbol{I_{3\times3}}\Delta t \\ \boldsymbol{O_{3\times3}} & \boldsymbol{I_{3\times3}} \end{bmatrix}, \boldsymbol{b} = \begin{bmatrix} \boldsymbol{O_{3\times1}} \\ \boldsymbol{g}\Delta t \end{bmatrix}$$

where $\boldsymbol{g}$ is gravity vector, which we assume to be constant.

Using the vision model described above, we can then model the observer. If $\boldsymbol{R_k}$ is the orientation and $\boldsymbol{r_k}$ the position of the eye frame at time k, the observed position and velocity of the ball $\boldsymbol{z_k}$ is:

$$\boldsymbol{z_k} = \boldsymbol{H_k x_k} + \boldsymbol{h_k} + \boldsymbol{v_k}$$

where $v_k$ is the observation noise defined above while $H_k$ and $h_k$ are the transformation from the spatial to the eye frame:

$$\boldsymbol{H_k} = \begin{bmatrix} \boldsymbol{R_k}^{\boldsymbol{T}} & \boldsymbol{O_{3\times3}} \\ \boldsymbol{O_{3\times3}} & \boldsymbol{R_k}^{\boldsymbol{T}} \end{bmatrix}, \boldsymbol{h_k} = \begin{bmatrix} \boldsymbol{R_k}^{\boldsymbol{T}} \boldsymbol{r_k} \\ \boldsymbol{O_{3\times3}} \end{bmatrix}$$

We use a standard Kalman filter to implement the Bayesian inference with internalized dynamics in the brain. For completeness, the predict-update filtering algorithm is given below.

- Predict

$$\boldsymbol{x_{k|k-1}} = \boldsymbol{F x_{k-1|k-1}} + \boldsymbol{b}$$
$$\boldsymbol{P_{k|k-1}} = \boldsymbol{F P_{k-1|k-1}} + \boldsymbol{b F^T}$$

- Update

$$\boldsymbol{y_k} = \boldsymbol{z_k} - \boldsymbol{H_k x_{k|k-1}} - \boldsymbol{h_k}$$
$$\boldsymbol{S_k} = \boldsymbol{H_k P_{k|k-1} H_k}^{\boldsymbol{T}} + \boldsymbol{A_k \Sigma_k}^{\boldsymbol{2}} \boldsymbol{A_k}^{\boldsymbol{T}}$$
$$\boldsymbol{K_k} = \boldsymbol{P_{k|k-1} H_k}^{\boldsymbol{T}} \boldsymbol{S_k}^{\boldsymbol{-1}}$$
$$\boldsymbol{x_{k|k}} = \boldsymbol{x_{k|k-1}} + \boldsymbol{K_k y_k}$$
$$\boldsymbol{P_{k|k}} = (\boldsymbol{I} - \boldsymbol{K_k H_k}) \boldsymbol{P_{k|k-1}}$$

where $A_k$ and $\Sigma_k$ are previously defined. Since we assume that the subject initially has no prior knowledge about the states, the initial value of the prior estimate covariance, $P_{0|0}$ is made sufficiently large and $x_{0|0}$ is chosen to be the zero vector.

# 3 Human Modeling

## 3.1 Full-Body Model

The full-body model has 75 bones with 165 degrees of freedom. This anatomical modeling makes it possible to generate human like motion; however, this introduces a new challenge in controlling the biomechanical system because of the kinematic redundancy. The skeleton is incorporated with 846 muscles. The detail of this model is explained in [2].

The skeleton is modeled as an articulated, multi-body dynamics system. All the vertebrae in the lumbar, thoracic, and cervical regions are modeled as individual rigid bodies with 3 degree-of-freedom joints.

The arms have clavicle, scapula, upper arm, ulnar, radius, hand, and the joint have 0, 3, 1, 1, 2 degrees of freedom, respectively. The clavicle is attached with the top rib, and the scapula is rigidly attached to the clavicle. The total number of degrees of freedom of the arm is 7.

# 4 Control

## 4.1 Full-Body Inverse Kinematics Control

We use a simple inverse kinematics approach. The input can be sequential target such as motion capture data, and we use inverse kinematics to compute the required angles of each joint. In our experiment, we give estimated target position as input and control the left arm accordingly.

## 4.2 Full-Body Muscle Control

We tried muscle control system. This experiment did not work well because of its high computational cost. This approach is more realistic way of control system. We would like to try machine learning approach to optimize the control system as a future project.

First, the desired accelerations are computed as follows: We iteratively update the joint angles in a gradient decent manner such that the difference of the current and target pose are minimized. The controller determines the required acceleration to reach the target at each time step using the following function:

$$\ddot{q} = k_p(q^* - q) + k_v(\dot{q}^* - \dot{q}), \tag{5}$$

where $k_p$ and $k_v$ determine the characteristic of the acceleration given the difference between the current and target positions, and also the difference of the velocity.

After computing the desired acceleration, we compute the generalized force for each joint by inverse dynamics. We use Featherstone's algorithm to efficiently compute the desired torque by the recursion algorithm.

Once we get the desired torque, we compute the muscle activation level. We use SNOPT library for the computation. The activation levels of each muscle is passed to the force computation system and the adequate muscle is calculated.

Then, we run forward dynamics to compute the acceleration of the system, and use first-order implicit Newton-Euler integration to simulate the character motion. The equations of motion of the skeletal system are written as

$$M(q)\ddot{q} + C(q,\dot{q}) = \tau + J^T f_e, \tag{6}$$

where, $M$ is mass matrix, $C$ accounts for forces such as the force from connecting tissues as well as Coriolis forces and centrifugal forces. $\dot{q}$ and $\ddot{q}$ are the state velocity and acceleration. $J$ is Jacobian matrix, which transforms applied external force to joint torque. This equation can be written as

$$\ddot{q} = \phi(q,\dot{q},\tau). \tag{7}$$

We use forward dynamics to compute $\phi$ by computing the $\ddot{q}$ from the generated torque with the muscle force. Then, we use the implicit Euler time integration method to solve the linearized equations of motion. We can compute velocity at the next step by solving

$$\dot{q}(t+h) - \dot{q}(t) = h\phi(q(t+h), \dot{q}(t+h), \tau). \tag{8}$$

The problem here is that this equation has the variable at the next time step on the right hand of the equation. We use first-order approximation and rewrite the equation as follows:

$$\begin{aligned} \delta\dot{q} &= h\left[\phi(q(t),\dot{q}(t),\tau) + \frac{\partial\phi}{\partial q}\delta q + \frac{\partial\phi}{\partial\dot{q}}\delta\dot{q}\right] \\ &= h\left[\phi(q(t),\dot{q}(t),\tau) + \frac{\partial\phi}{\partial q}h(\dot{q}(t)+\delta\dot{q}) + \frac{\partial\phi}{\partial\dot{q}}\delta\dot{q}\right] \end{aligned} \tag{9}$$

Thus, we can compute the velocity at next time step. We get position at the next time step with the explicit time integration of velocity.

## 5 Results

The result of our experiment can be found here.
(http://www.cs.ucla.edu/ nakada/page1/page10/index.html)

As it can be seen from the video, we achieved vision-driven body control for a comprehensive biomechanical human model.

## 6 Conclusion and Future Work

The contribution of this research is the control of the left arm with the full-body biomechanical model driven by visual information. We used a simple edge detection algorithm to detect the target object, and used bounding box to estimate the target size and the location. Then we estimate the tareget depth with kalman filter and reach the target with inverse kinematics.

Inverse kinematics approach showed realistic motion of the character. Our kalman filter worked to filter out the noises from the visual information. The noise mainly co mes from the computation of the depth with stereo vision. It is even difficult for human to accurately estimate the target distance from eyes, and it is reasonable for the virtual character to have some error. We use the output from kalman filter to control the arm so that the target of the control can be smooth enough to achieve natural motion.

Our muscle based control did not work well in our experiment. The most challenging problem comes from the computational complexity. The control space is too big to compute in real time. We plan to apply machine learning technique and learn with training samples so that we can efficiently compute the muscle activation and simulate in real time. This is our future work. [3] proved that off-line neural network learning works for the biomechanical neck model. We would like to take this approach in our full body model.

As another future work, we would also like to implement different detection algorithms in this full-body model. It can currently detect an object; however, the character does not know what the detected object is. We would like to achieve object recognition by implementing techniques introduced by [4]. This feature would make the virtual character more realistic and useful.

## References

[1] J Canny. A computational approach to edge detection. *IEEE transactions on pattern analysis and machine intelligence*, 8(6):679–98, June 1986.

[2] Sung-Hee Lee, Eftychios Sifakis, and Demetri Terzopoulos. Comprehensive biomechanical modeling and simulation of the upper body. *ACM Transactions on Graphics*, 28(4):1–17, August 2009.

[3] Sung-hee Lee and Demetri Terzopoulos. Heads Up ! Biomechanical Modeling and Neuromuscular Control of the Neck. 1(212):1188–1198, 2006.

[4] D.G. Lowe. Object recognition from local scale-invariant features. pages 1150–1157 vol.2, 1999.

[5] Schlumberger Palo and Palo Alto. Snakes : Active Contour Models. 331:321–331, 1988.

[6] SH Yeo, Martin Lesmana, DR Neog, and DK Pai. Eyecatch: simulating visuomotor coordination for object interception. *ACM Transactions on Graphics ...*, 31(4):1–10, 2012.