Created by He Entong

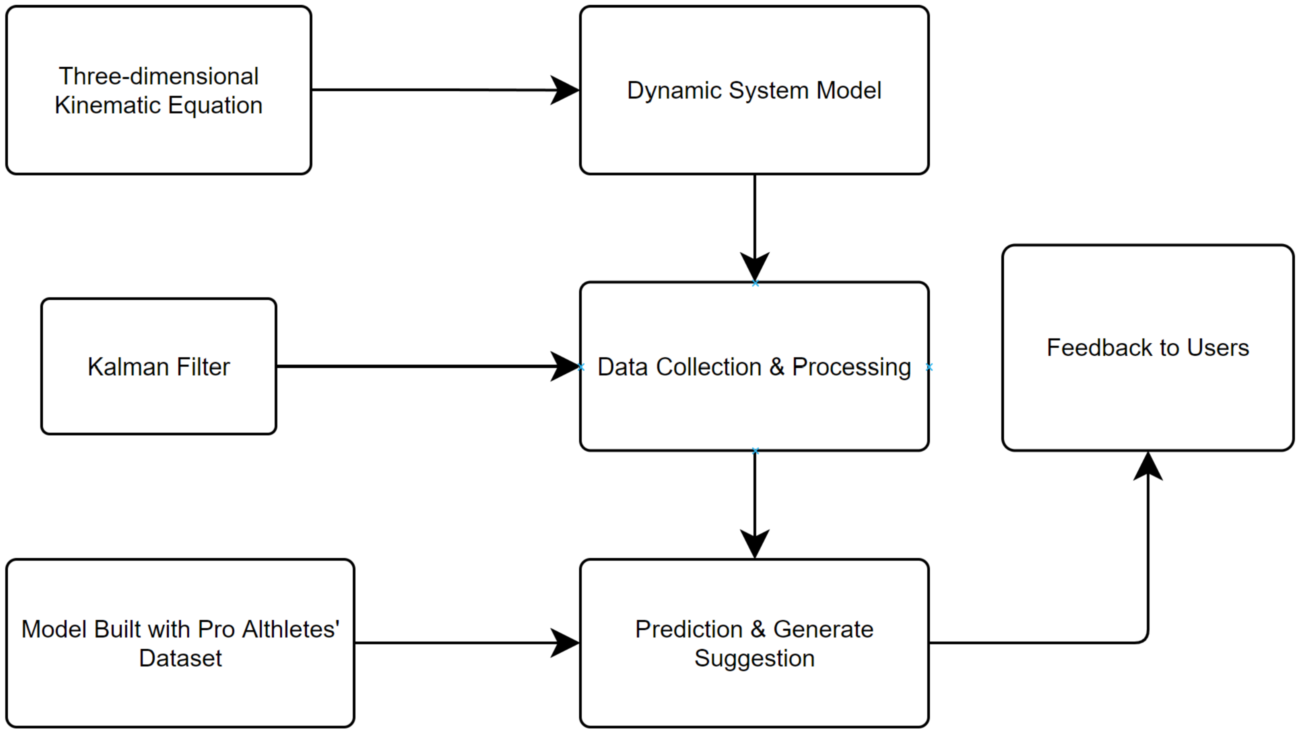
Contents

1 Introduction

1.1 Project Detail

This article is used for ENGG1320 group project: AI Coaching System. We introduced a method to calculate the limb actions of basketball players with wearable device (bracelet integrated with gyroscope, accelerometer and velocity detector and action-tracing gloves) and use machine learning to generate feedback for users.

1.2 Algorithm Framework



2 Kalman Filter (Only for Hardware Not Integrated with A Filter)

Our model requires Bayesian model to build up the detection. Thus we apply Kalman filter on the dataset. The following gives a brief derivation.

2.1 Derivation

Two possible noise are the white noise $w_k \sim N(0, \sigma^2_{w_k})$ and measurement noise v_k. Then for real data and data measured,

$$x_{k+1} = \Phi x_k + w_k \quad z_k = Hx_k + v_k \tag{1}$$

Assume for data estimation, the prior prediction is $\hat{x}'_k$, and actual estimation is $\hat{x}_k$. Estimation bias is $e_k = x_k - \hat{x}_k$, and one assumption is that prior model bias is proportional to measurement bias, i.e.

$$\hat{x}_k - \hat{x}'_k = K_k(z_k - H\hat{x}'_k) \tag{2}$$

where K_k is the Kalman gain. The filter tends to maximize the sum of variance of estimation bias. We first consider bias covariance matrix

$$\begin{aligned} E(e_k e_k^T) = P_k = E[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T] \\ E[(x_k - \hat{x}'_k - K_k z_k + K_k H \hat{x}_k)(x_k - \hat{x}'_k - K_k z_k + K_k H \hat{x}_k)^T] \\ = (I - K_k H) E[(x_k - \hat{x}'_k)(x_k - \hat{x}'_k)^T](I - K_k H)^T + K_k E(v_k v_k^T) K_k^T \end{aligned} \tag{3}$$

Define $P'_k = E[(x_k - \hat{x}'_k)(x_k - \hat{x}'_k)^T] = E(e'_k e'^T_k), E(v_k v_k^T) = R$ as the prior covariance matrix. Then

$$P_k = (I - K_k H) P'_k (I - K_k H)^T + K_k R K_k^T \tag{4}$$

The model tends to minimize the variance of estimation bias (trace of covariance matrix), according to which coefficient K_k is determined, that is

$$\begin{aligned} &\arg\min_{K_k} \boldsymbol{tr}(P_k) \to \frac{\partial}{\partial K_k} \boldsymbol{tr}(P_k) = 0 \\ &\longrightarrow -2(HP'_k)^T + 2K_k(HP'_k H^T + R) = 0 \\ &\longrightarrow K_k = P'_k H^T (HP'_k H^T + R)^{-1} = K_k^* \end{aligned} \tag{5}$$

then

$$\begin{aligned} \min \boldsymbol{tr}(P_k) &= \boldsymbol{tr}(P'_k) - 2\boldsymbol{tr}[K_k H P'_k] + T\left[K_k(HP'_k H + R)K_k^T\right]\Bigg|_{K_k = K_k^*} \\ &= (I - K_k^* H) P'_k \end{aligned} \tag{6}$$

Then the prior prediction bias $e'_{k+1} = x_{k+1} - \hat{x}'_{k+1}$. Based on previous data, prior prediction can be predicted as

$$\hat{x}'_{k+1} = \Phi \hat{x}_k \tag{7}$$

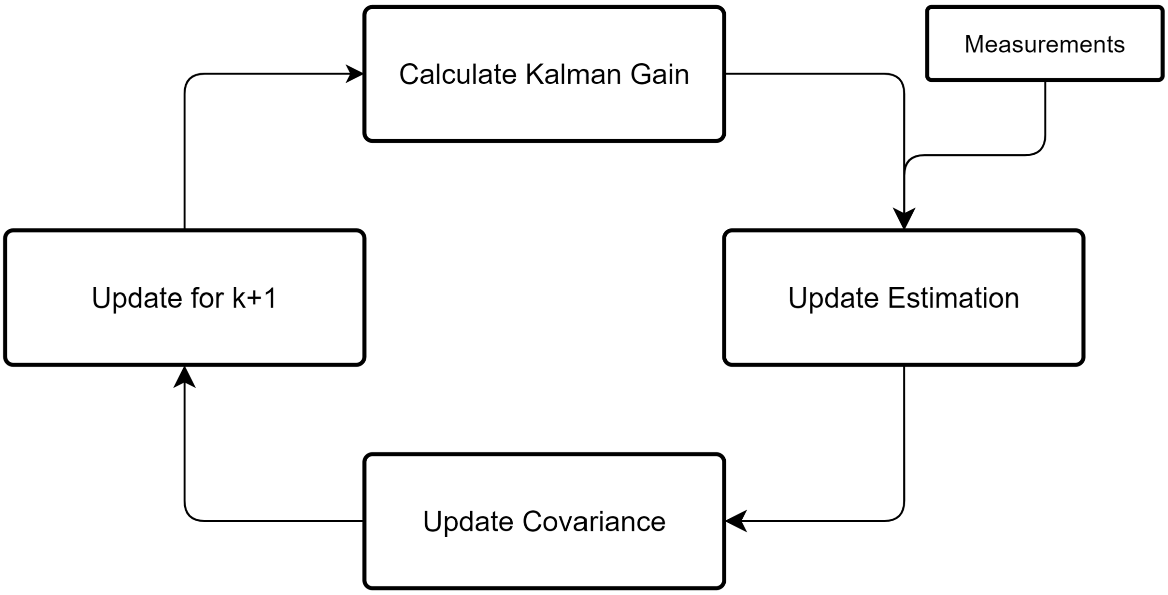
Thus the prior prediction bias can be calculated as follows

$$\begin{aligned} e'_{k+1} &= x_{k+1} - \hat{x}'_{k+1} \\ &= (\Phi x_k + w_k) - \Phi \hat{x}_k \\ &= \Phi e_k + w_k \end{aligned} \tag{8}$$

Notice that the bias and noise are mutually independent, hence the prior prediction bias matrix

$$\begin{aligned} P'_{k+1} &= E(e'_{k+1} e^T_{k+1}) \\ &= E\left[(\Phi e_k + w_k)(\Phi e_k + w_k)^T\right] \\ &= \Phi E(e_k e_k^T)\Phi^T + E(w_k w_k^T) = \Phi P_k \Phi^T + Q_w \end{aligned} \tag{9}$$

2.2 Loop Procedure



Algorithm 1 Kalman Filter

Require: sampling period T_s, initial state P'_0

while $k \in T_s$ **do**

Kalman Gain: $K_k = P'_k H^T (H P'_k H^T + R)^{-1}$

Update Estimate: $\hat{x}_k = \hat{x}'_k + K_k(z_k - H\hat{x}'_k)$

Update Covariance: $P_k = (I - K_k H)P'_k$

Project to Next Loop:

$$\hat{x}'_{k+1} = \Phi \hat{x}_k$$
$$P_{k+1} = \Phi P_k \Phi^T + Q_w$$

end while

3 Physical Model

The following is the physical model to be applied in the motion detection algorithm. Physical quantities we require include the body velocity v_b, acceleration a_b, limb angular velocity ω_b and attitude angles (roll angle γ, pitch angle θ).

3.1 Kinematic Equation

We first consider kinematic equation in limb movement. Acceleration a_b gives

$$\begin{aligned} a_b &= \frac{\partial}{\partial t} v_b = \frac{\partial}{\partial t}(v_\theta \hat{\theta} + v_r \hat{r}) \\ &= (\dot{v_\theta} + \dot{v_r}) + (v_\theta \frac{\partial \hat{\theta}}{\partial \theta}\frac{\partial \theta}{\partial t} + v_r \frac{\partial \hat{r}}{\partial \theta}\frac{\partial \theta}{\partial t}) = (\dot{v_\theta} + \dot{v_r}) + \omega(-v_\theta \hat{r} + v_r \hat{\theta}) = \dot{v_b} + \omega \times v_b \end{aligned} \tag{10}$$

Accelerometer measurement f_b and body acceleration a_b has relationship

$$f_b = g_b + a_b \tag{11}$$

Assume in accelerometer measurement, acceleration a_b has relationship

$$\dot{a_b} = \eta a_b + w_1 \tag{12}$$

where Gaussian noise $w_1 \sim N(0, \sigma_{w_1}^2)$, and rational coefficient η is a constant.

3.2 Stochastic Model

Assume that we combine the limb motion together to form a matrix X_1, where

$$X_1 = \begin{bmatrix} v_b \\ a_b \end{bmatrix} \tag{13}$$

and it holds for time-shifting relationship as follows, where gyroscope zero shift w_g has standard deviation σ_{w_g}.

$$\dot{X_1} = \begin{bmatrix} \dot{v_b} \\ \dot{a_b} \end{bmatrix} = \begin{bmatrix} a_b - \omega_b \times v_b + w_g \\ \eta a_b + w_1 \end{bmatrix} = \begin{bmatrix} -[\omega_b]_\times & I_3 \\ 0_3 & \eta I \end{bmatrix} X_1 + \begin{bmatrix} w_g \\ w_1 \end{bmatrix} = A_1 X_1 + W_1 \tag{14}$$

During sampling period $T_s \in \mathcal{T}$, we have state transition equation

$$\begin{aligned} v_b^{(k+1)} &\approx v_b^{(k)} + \dot{v_b}^{(k)} \Delta t \to v_b^{(k+1)} \approx \left(I - [\omega_b]_\times T_s + R_a^v\right) v_b^{(k)} + w_g T_s \\ a_b^{(k+1)} &\approx a_b^{(k)} + \dot{a_b}^{(k)} \Delta t \to a_b^{(k+1)} \approx (I + \eta I T_s) a_b^{(k)} + w_1 T_s \end{aligned} \tag{15}$$

Then the noise covariance matrix, given that the measurement and noise have no correlation, is

$$Q_1^{(k)} = \begin{bmatrix} Q_{w_g}^{(k)} & 0_3 \\ 0_3 & Q_{w_1}^{(k)} \end{bmatrix} = \begin{bmatrix} \sigma_{w_g}^2 T_s^2 [v_b]_\times^{(k)} \left([v_b]_\times^{(k)}\right)^T & 0_3 \\ 0_3 & \sigma_{w_1}^2 T_s^2 I_3 \end{bmatrix} \tag{16}$$

Assume the noise can be neglected in the iteration of dynamic system, then the state equation in respect of time for X_1 is

$$\dot{X_1} = A_1(T_s) X_1 \to X_1^{(k+1)} = \exp\left(A_1^{(k)} T_s\right) X_1^{(k)} + W_1^{(k)} \tag{17}$$

Zhaoying Zhou et al. (2004) developed equation

$$\begin{bmatrix} \dot{g_x} \\ \dot{g_y} \\ \dot{g_z} \end{bmatrix} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \begin{bmatrix} g_x \\ g_y \\ g_z \end{bmatrix} \longrightarrow \dot{g_b} = [\omega_b]_\times g_b \tag{18}$$

Like wise, the recursive relationship and covariance matrix can be derived as

$$g_b^{(k+1)} = \exp\left([\omega_b]_\times^{(k)} T_s\right) g_b^{(k)} + W_2^{(k)} \tag{19}$$

where W_2 is corresponding process noise. Likewise,

$$\begin{aligned} g_b^{(k+1)} \approx g_b^{(k)} + \dot{g}_b^{(k)} \Delta t &= g_b^{(k)} + [\omega_b]_\times (g_{b\ \text{real}}^{(k)} + g_{b\ \text{bias}}^{(k)} + g_{b\ \text{noise}}) T_s \\ &\approx (I + [\omega_b]_\times T_s) g_b^{(k)} + [\omega_b]_\times g_{b\ \text{noise}} T_s \end{aligned} \tag{20}$$

Then the noise covariance equation

$$Q_2 = \sigma_g^2 T_s^2 [g_b]_\times \left([g_b]_\times\right)^T \tag{21}$$

3.3 Measurement Model

Assume that the data (tri-axis velocity v_m and body total acceleration f_m) we collect from wearable device have Gaussian noise $\nu_1 \sim N(0, \sigma_{\nu_1}), \nu_2 \sim N(0, \sigma_{\nu_2})$ respectively. Then denote the state matrix as Y.

$$Y = \begin{bmatrix} v_m \\ f_m \end{bmatrix} = \begin{bmatrix} v_b + \nu_1 \\ a_b + g_b + \nu_2 \end{bmatrix} = \begin{bmatrix} I_3 & 0_3 & 0_3 \\ 0_3 & I_3 & I_3 \end{bmatrix} \begin{bmatrix} v_b \\ a_b \\ g_b \end{bmatrix} = H \begin{bmatrix} X_1 \\ g_b \end{bmatrix} + \nu \tag{22}$$

Take the augmented state matrix $X = \begin{bmatrix} X_1 \\ g_b \end{bmatrix}$ and augmented noise matrix $W = \begin{bmatrix} W_1 \\ W_2 \end{bmatrix}$, then

$$Y = HX + \nu \to Y^{(k+1)} = H^{(k)} X^{(k)} + \nu^{(k)} \tag{23}$$

$$X^{(k+1)} = \begin{bmatrix} \exp\left(A_1^{(k)} T_s\right) & 0_3 \\ 0_3 & \exp\left([\omega_b]_\times^{(k)} T_s\right) \end{bmatrix} X^{(k)} + W^{(k)} = \Phi^{(k)} X^{(k)} + W^{(k)} \tag{24}$$

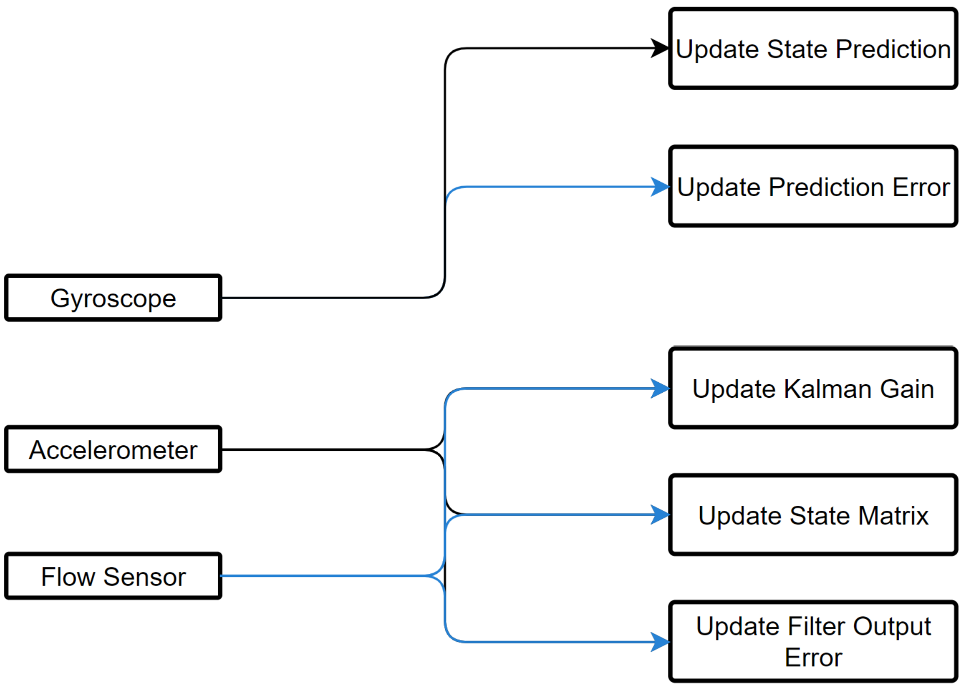
$$Q^{(k)} = \begin{bmatrix} Q_1^{(k)} & 0_{6\times3} \\ 0_{3\times6} & Q_2^{(k)} \end{bmatrix} \tag{25}$$

Measurement noise covariance R can be calculated as

$$R^{(k)} = \nu^{(k)} \left(\nu^{(k)}\right)^T \tag{26}$$

3.4 Filtering Process

Assume that P is the state prediction error, then the filtering process is shown as



Algorithm 2 Data Measurement Filtering

Require: sampling period T_s, initial state P_0'

while $k \in T_s$ **do**

Update state prediction: $(X^{(k+1)})' = \Phi^{(k)} X^{(k)}$

Update prediction error: $(P^{(k+1)})' = \Phi^{(k)} P^{(k)} (\Phi^{(k)})^T + Q^{(k)}$

Update Kalman gain: $K^{(k+1)} = (P^{(k+1)})'(H^{(k)})^T \left(H^{(k)}(P^{(k+1)})'(H^{(k)})^T + R^{(k)}\right)^{-1}$

Update state matrix: $X^{(k+1)} = (X^{(k+1)})' + K^{(k+1)}(Y^{(k+1)} - H^{(k)}(X^{(k+1)})')$

Update covariance: $P^{(k+1)} = (I - K^{(k+1)} H^{(k)})(P^{(k+1)})'$

end while

3.5 From Kinematic Quantities to Attitude

We use Euler's angle to depict attitude of limbs. Roll and pitch angles (γ, θ) can be measured by the gravitational acceleration g_b we obtained from the model. Assume no yaw angle occurs, then

$$\begin{aligned} \boldsymbol{g}_b = \boldsymbol{Z}_\gamma \boldsymbol{Z}_\theta \boldsymbol{g} &= \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} \\ &= \begin{bmatrix} \sin\theta\cos\gamma \\ \sin\gamma\sin\theta \\ \cos\theta \end{bmatrix} g \end{aligned} \tag{27}$$

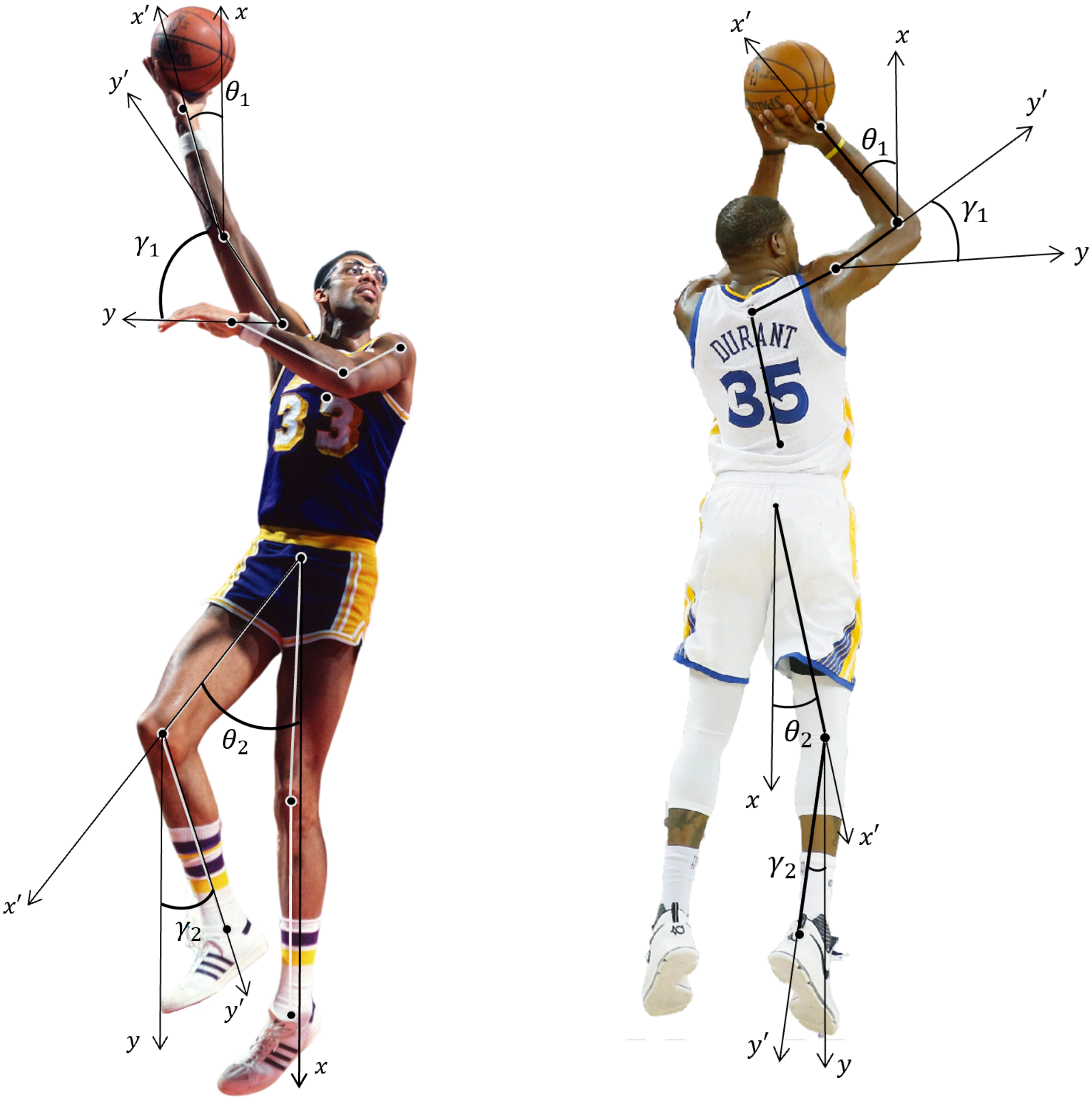
Then we can come up with the attitude from the body reference frame gravitational acceleration, where

$$\begin{cases} \gamma = \arctan \dfrac{g_{by}}{g_{bx}} \\ \theta = \arctan \left(\dfrac{g_{by}}{\sin\gamma g_{bz}} \right) \end{cases} \tag{28}$$

4 Data Analysis

4.1 Model of Users' Motion

We establish the state model $\boldsymbol{S}$ for motion of players.



Data considered in analysis section includes the roll angle and pitch angle of players' limbs. Upper limb angles are denoted as $(\boldsymbol{\theta}_1, \boldsymbol{\gamma}_1)$, and thigh angles are denoted as $(\boldsymbol{\theta}_2, \boldsymbol{\gamma}_2)$. Wearable devices are adjusted to two independent coordinate respectively. Average angular velocity $\bar{\boldsymbol{\omega}}$ can be measured by

$$\bar{\boldsymbol{\omega}} = \frac{1}{\sum_{T_s} \Delta t} \sum_{t \in T_s} \boldsymbol{\theta}(t) \tag{29}$$

Assume during time period (t_1, t_2), positive total acceleration is detected, then the maximum height h from the ground can be measured by

$$h = \frac{1}{2} g \left(\sum_{t_1}^{t_2} \Delta t \right)^2 \tag{30}$$

Under polar coordinate, we depict the comparative position of user with the basket with (β, r). Then the complete characteristic vector $\boldsymbol{S}$ is

$$\boldsymbol{S} = \begin{bmatrix} \boldsymbol{\theta}_1 & \boldsymbol{\gamma}_1 & \boldsymbol{\theta}_2 & \boldsymbol{\gamma}_2 & \bar{\boldsymbol{\omega}}_1 & h & \beta & r \end{bmatrix}^T \tag{31}$$

where

$$\boldsymbol{\theta}_1 = \begin{bmatrix} \theta_{\text{left upperlimb}} & \theta_{\text{right upperlimb}} \end{bmatrix} \tag{32}$$
$$\boldsymbol{\gamma}_1 = \begin{bmatrix} \gamma_{\text{left upperlimb}} & \gamma_{\text{right upperlimb}} \end{bmatrix} \tag{33}$$
$$\boldsymbol{\theta}_2 = \begin{bmatrix} \theta_{\text{left thigh}} & \theta_{\text{right thigh}} \end{bmatrix} \tag{34}$$
$$\boldsymbol{\gamma}_1 = \begin{bmatrix} \gamma_{\text{left thigh}} & \gamma_{\text{right thigh}} \end{bmatrix} \tag{35}$$

4.2 Data Analysis Method: Support Vector Machine

The user shooting outcome can be mapped into label space $\mathcal{H}$, where

$$\mathcal{H} = \begin{cases} -1 & \text{field goal miss} \\ 1 & \text{field goal made} \end{cases} \tag{36}$$

We use Support Vector Machine to study the motions of professional athletes. To avoid non-linear dataset, radius basic function is applied as the kernel function, where

$$K(x, y) = \exp\left(-\frac{||x-y||^2}{2\sigma^2}\right) \tag{37}$$

Penalty factor C is set comparatively low to achieve higher generalization. SVM training will find solution of following target

$$\begin{aligned} &\arg\min_{\boldsymbol{\alpha}} \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) - \sum_{i=1}^{N}\alpha_i \\ &\text{s.t.} \sum_{i=1}^{N}\alpha_i y_i = 0,\ 0 \le \alpha_i \le C \end{aligned} \tag{38}$$

with optimal multiplier $\boldsymbol{\alpha}^*$ obtained, the weight $\boldsymbol{w}$ and bias b can be calculated

$$\boldsymbol{w}^* = \sum_{i=1}^{n}\alpha_i^* y_i \boldsymbol{x}_i \tag{39}$$

$$y_j(K(\boldsymbol{w}^*, \boldsymbol{x}_j) + b^*) = 1 \to b^* = y_j - \sum_{i=1}^{n}\alpha_i^* y_i K(\boldsymbol{x}_i, \boldsymbol{x}_j) \tag{40}$$

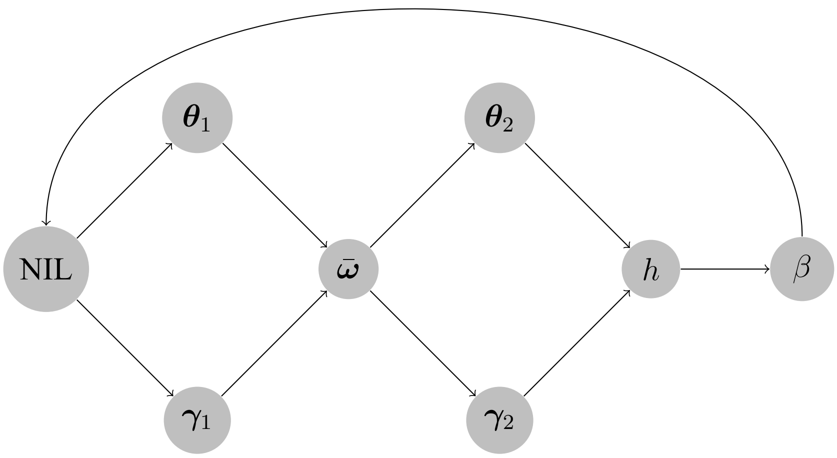
We use sklearn library integrated in Python (prototype LIBSVM in C++) to complete the training process. Weight vector and bias will be stored in the *.m file to be used for data analysis later on. The update on the training app will include maintenance on shooting database.

4.3 Feedback Algorithm

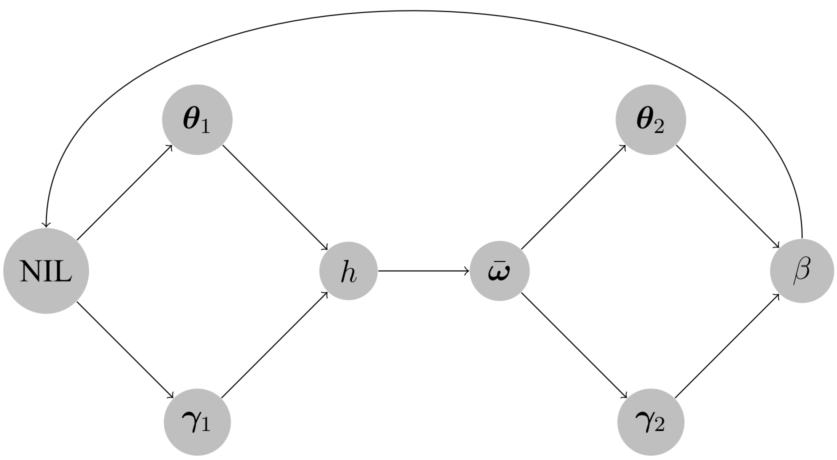
Once the weight vector and bias is studied, it can be used in giving feedback to users. Assume that the data collected from user is $\boldsymbol{S}$, then the decision-making function is

$$f(\boldsymbol{S}) = sign(K(\boldsymbol{w}, \boldsymbol{S}) + b) \tag{41}$$

We tend to make the inner quantity as large as possible, which increases the possibility to make field goal according to the database. Since all the physical quantities take positive values, we use an intuitive algorithm to generate the feedback: increase the parameter with positive weight, and decrease the parameter with negative weight. We take order to optimize the shooting motion. With short-distance shot (inside the free throw line), we suggest two-motion shots. So bounce height will be considered more important. By contrast, in long-distance one, we suggest one-motion shots and put more priority to upper limb and thigh attitude. Assume the threshold value for distance is r_t, then the following gives the graph of our optimizing order.
With $r < r_t$ (two-motion shot),



With $r \geq r_t$ (one-motion shot),



In every feedback process, the program will search for optimization in the graph above using BFS. It continually searches for positive weight nodes and negative weight nodes, and use step size η which is uniquely set up for each node. Every time a node v is visited, its corresponding step size η_v will be added/deduced according to its weight.

Algorithm 3 BFS($\boldsymbol{S}$)

Require: priority graph $G(V, E)$

intialize feedback vector $\boldsymbol{f} = \mathbf{0}$, starting vertex $s = G$.NIL, queue Q
$Q.enqueue(s)$
while $sign(K(\boldsymbol{w}, \boldsymbol{S} - \boldsymbol{f}) + b) \neq 1$ **do**
  $u \leftarrow Q.deque$
  for $v \in u.adj$ **do**
    if $\boldsymbol{w}.v \geq 0$ **then**
      $\boldsymbol{f}.v \leftarrow \boldsymbol{f}.v + \eta_v$
    else
      $\boldsymbol{f}.v \leftarrow \boldsymbol{f}.v - \eta_v$
    end if
    $Q.enqueue(v)$
  end for
end while
return $\boldsymbol{f}$

With the feedback vector, it is easy to generate the feedback for users literally and schematically.