Master Thesis

Human Motion Prediction Using 2D Person Pose Estimation on Unstable Data with RGB Camera

March, 2020

Yunus Andi Prademon

Human Interface Laboratory

Division of Information Engineering

Graduate School of Engineering

Mie University

Abstract

The development of the autonomous system in the world is growing rapidly, it is becoming important to solve many problems. In case of human-machine interaction, machines such as an autonomous car or a robot that works in human living environment need to know human's future motion for its moving trajectories. An autonomous system needs to know all possible human behaviors to estimate the future motion of human. However, our technology is still too far to remember all human behavior of movement which is unique by personality. Even though, in order to advance these trajectories, a research needs to be performed as soon as the technologies is growing itself.

Some of the previous works were using the Kinect RGB-D camera which has the depth sensor that could be used to provide the pose of human body.

This research uses the RGB camera as the other option that we can rely on. Currently, RGB-D camera is not widely available in many devices. We realize that the pose estimation which has been obtained from the RGB camera is not as precise as RGB-D camera yet. It is still reliable enough when we can optimize the data as an obstacle we need to get through. We propose the system to obtain the human body motion prediction by using regular digital RGB camera including a smartphone camera or even a surveillance camera. We set a goal to predict 1 second ahead of the motion, and 30 fps videos have been prepared which include simple motions such as hand gesture and walking movement.

We used OpenPose library from OpenCV to extract features of a human body pose including 14 points. Since OpenPose estimation is not always precise as we expected and to minimize the estimation error of the OpenPose we restricted the image area to perform human pose estimation using YOLOv3. We input distance and direction which are calculated from the features by comparing two consecutive frames into Recurrent Neural Network Long Short-Term Memory (RNN-LSTM) model and Kalman Filter. For the evaluation, we compare the result by the distance from the prediction result to the ground truth which is the position of the node after 1 second in the video and group the distance with value that are

lower than 1.8% of the diagonal frame size, we called it the successful percentage of prediction. As the results, Kalman Filter reached 93% in average, and RNN-LSTM reached 75% in average on our dataset. While Kalman Filter reached 77% in average, and RNN-LSTM reached 52% in average on CMU dataset. Mostly, Kalman Filter show better estimate accuracy than RNN-LSTM and based on the human motions, motion such as hand gesture and moving to the right side are easier than more complex motion like hand gesture and moving to the left side. We confirmed the validity of RGB-camera based method in the simple human motion case from the result, and we conclude that this is an important step to realize the prediction of more complex human motion.

Contents

Chapter 1

Introduction

Human pose estimation has been an interesting task to be be developed with many actual application that can be done. Several researches applied the human pose estimation to develop the augmented reality and computer game development. In this study, human pose estimation is used to develop the human motion prediction with RGB camera. In this section, the research's background and objectives are explained. As well as the related works with human motion prediction have been reviewed for the discussion about the relations and differences point of interest between the current and the past works.

1.1 Research Background

People constantly interact with everything around them, such as human to human interaction as well as human to machine interaction. We have been dreaming about robots that could coexist with humans, and now many people are competing to create the most reliable autonomous machines such as auto-driving car and auto-moving industrial robot. However, creating such a reliable autonomous machine is not an easy accomplishment, there are plenty of problems to solve, such as preventing an auto-driving car collide with another car, or even preventing it to run over a human being. If the auto-driving car cannot predict the human movement, the problem persists. With this in mind, a system to predict human motion is needed. On the other hand, human motion as the object is difficult to predict due to the countless motion in human behaviors, as well as the differences of the individual behaviors.

In recent years, Recurrent Neural Network (RNN) has been used in many cases for prediction including human motion prediction with high accuracy [1, 2, 3]. Since the human behaviors are individually unique and varied, we need a long term prediction algorithm. Recurrent Neural Network - Long Short Term Memory (RNN-LSTM) provides the long term

prediction and short term prediction based on its memory to save the values of behavior with high accuracy [4, 5]. On the other hand, Kalman Filter that has been used in many fields of researches and problems to estimate the movement and measurement with noise has shown reliable results. Kalman Filter is an efficient recursive filter that estimates the internal state of a linear dynamic system from a series of noisy measurements. Kalman Filter has also been used in some applications such as short-term forecasting and the analysis of life lengths from dose-response experiments [6].

Due to the countless of human behavior, predicting all of human motions are really hard task. With the current technology, this work seems impossible to be done. However, in this research we set the scope with only limited duration and motions that will be predicted. By this limitation, we can conduct the human motion part by part. Even though we still are not sure how many memory and GPU to process all of the motions. In this research, we perform a research on Intel Xeon E5-2665 and Nvidia GeForce GTX 1080 8GB.

1.2 Related Works

Some researches develop their systems with data from the RGB-D camera since it has depth parameter for human pose estimation [1, 7]. RGB-D camera such as Kinect camera can estimate human body parts precisely. However, this type of camera does not available in many places. Our main task of interest is human motion prediction with focusing on unstable data obtained from human pose estimation by RGB camera. Even though, the obtained data are not always give pose precisely of human body parts as shown in Fig. 1.1. These data give us another problem to solve. In this research, we proposed a method to examine the unstable data can be used for human motion prediction.

A research has been performed for human motion prediction with RGB camera [8]. They focused on human motion forecasting of sports activity especially for safe martial arts such as boxing, karate or taekwondo. As a result, they obtained 0.5 second of human motion prediction by forecasting 15 frame step in a 30fps video. Nonetheless, they did not show the accuracy of the prediction.

1.2.1 RGB-D camera based human motion prediction

RGB-D camera such as Kinect from Microsoft provides the precise human pose estimation by computing the depth parameter which is obtained from the depth sensor in the camera. In case of the human motion prediction, the data that will be processed is the pose of human,

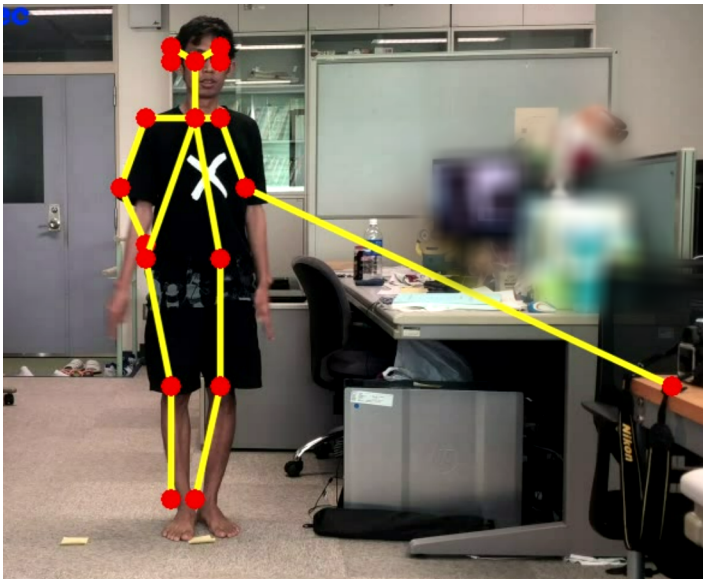


Fig.1.1: Estimation error in OpenPose

thus, kinect camera is an equitable option as the input. A research have been performed for predicting the human motion [9] with RGB-D camera. Yujiao Cheng *et al.* [9] use the recursive least square parameter adaptation algorithm (RLS-PAA) to predict the data which are provided from the kinect camera dataset, CMU dataset, time invariant dataset, and time varying dataset. All of the datasets are used with motion capture data. These dataset are provided as the verification for the performance of the RLS-PPA. As the result, they obtained the least Mean Squared Estimation Error (MSEE value) on every datasets by neural network with recursive least square parameter adaptation algorithm (RLS-PAA) method.

1.2.2 RGB camera based human motion prediction

Yongyi Tang *et al.* [7] use the dataset from Human 3.6m dataset which is the largest human motion dataset for 3D body pose analysis. In their research, they proposed a new method for the prediction process. They verified their prediction method which is the modified highway unit (MHU) by comparing with ERD [10], LSTM3LR [11], Res-GRU [1]. As the result, mostly their method performed better than the other method. However, on some motion such as walking, posting, and taking picture the prediction results have shown mostly fewer accuracy than the other method.

Erwin Wu, *et al.* [8] performed a research with concern of real-time human motion forecasting using RGB camera. Their research focused on sport activity such as Taekwondo. They applied the real-time human motion forecasting with VR (Virtual Reality) head-mounted dis-

play. They obtained 0.5 second prediction.

1.3 Research Objective

The work of human motion prediction is a challenging task since even a human itself cannot do the same task when the development of technology is trying to reach it. This research is conducted to solve the safety problem on autonomous system which its development covers the living environment friendly machines. The main objective of this research are :

1. To study the method of human motion prediction with the comparison of deep learning and statistical Kalman Filter.
2. To perform the more complex sample on usage of deep learning and Kalman Filter

Chapter 2

Neural Network Techniques

Human pose estimation has been an interest issue to be be developed with many actual application that can be done. Several researches applied the human pose estimation to develop the augmented reality and game development. In this study, human pose estimation is used to develop the human motion prediction with RGB camera. In chapter 2.2, related works with human motion prediction have been reviewed for the discussion about the relations and differences point of interest between the current and the past works.

2.1 Deep Neural Network

Deep neural network method has been used for many purposes to solve various problems considering its performace and high adaptivity. Sreelekshmy Selvin [12] performed a research to predict the stock price using LSTM, RNNs and CNN-sliding window model. They compare these deep learning architectures one by one for the stock market dataset to see the response by each model. As the results, they conclude CNN architecture is capable of identifying the changes in trends and CNN is identified as the best model for their methodology. Even though, this research is not so close to our research which is predicting the sequence data by behavior of the previous data. As an example for the similar subject of research, Alex Graves [2] explained about RNN-LSTM for more detail in his paper. He implemented RNN-LSTM in some problems that include text prediction, handwriting prediction, and handwriting systhesis.

Recurrent Neural Networks (RNNs) is one of the deep learning architecture that has been widely used for generating sequence data such as text [13] and computer vision [8]. RNNs can be trained for sequence generation by processing real data sequences one step at a time and predicting what comes next [2]. Recurrent network in principle use their feedback con-

nections to store representation of recent input events in form of activations (”short-term memory”) [5]. However, RNNs has its limitation of memory. the standard RNN model is unable to store the information for a long sequences. This limitation of RNNs can be referred to an analogy of a short-term amnesia such as Alzheimer on human brain. For some problem, with a few sequences of data RNNs will perform well, but when the data are accumulated RNNs can only store a limited sequence of data and will forget the long past data. This problem makes RNNs is not suitable to solve the very complex problem like human motion. All recurrent neural networks have the form of a chain of repeating modules of neural network. In standard RNNs, this repeating module will have a very simple structure, such as a single tanh layer for the activation function. As shown on Fig. 2.1, the repeating module in a standard RNN contains a single layer.

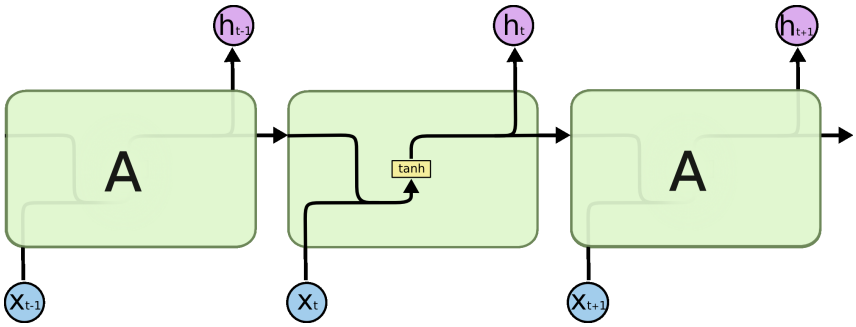
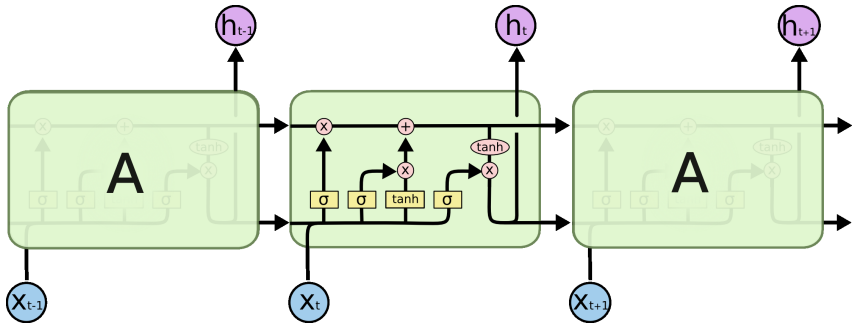
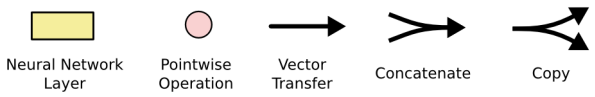


Fig.2.1: RNN architecture



(a) RNN-LSTM architecture



(b) Legends

Fig.2.2: RNN-LSTM structure

2.2 RNN - LSTM

The development of RNNs is solving the classic problem of the memory limitation. LSTM (Long Short Term Memory) is an extended RNNs architecture that designed to be better at storing and accessing than a standard RNNs. LSTM was introduced by Hochreiter & Schmidhuber (1997) [5], and were refined and popularized by many people in following work. As well as the standard RNN model, LSTMs also have this chain like structure of the repeating module with 4 layers function in a sequence as shown in Fig. 3.4.

2.3 OpenPose

OpenPose represents the first real-time multi-person system to jointly detect human body, hand, facial, and foot keypoints (in total 135 keypoints) on single images [14]. For the 2D real-time multi-person keypoint detection, OpenPose provides 15 or 18 or 25 body/foot keypoints estimation based on the dataset keypoints. The input for OpenPose are image, video, webcam, flir/point grey and IP camera. As the output, OpenPose generate the basic image with keypoints display/saving on several format like PNG, JPG, and AVI and/or keypoints as array class. The available keypoints from OpenPose by COCO output format consists nose, neck, shoulders, elbows, wrists, hips, knees, ankles, eyes, and ears, and background. While MPII output format consists head, neck, shoulders, elbows, wrists, hips, knees, ankles, chest, and background [16].

2.4 YOLOv3

You only look once (YOLO) is an object detection system targeted for realtime processing. Fast YOLO is the fastest general-purpose object detector in the literature and YOLO pushes the state-of-the-art in real-time object detection. YOLO also generalizes well to new domains making it ideal for applications that rely on fast, robust object detection [15].

Chapter 3

Dataset

Several works have been done preparing the dataset that contain human motion. These dataset explained in the following.

3.1 Our Dataset

For this research, we prepared 4 videos that contain simple motions such as hand gestures, moving aside, and sitting motion. The video with the sitting motion has 13 seconds of duration and the video without sitting motion has 7 seconds of duration. This is the first step on our research to realize the human motion prediction.

3.2 CMU Dataset

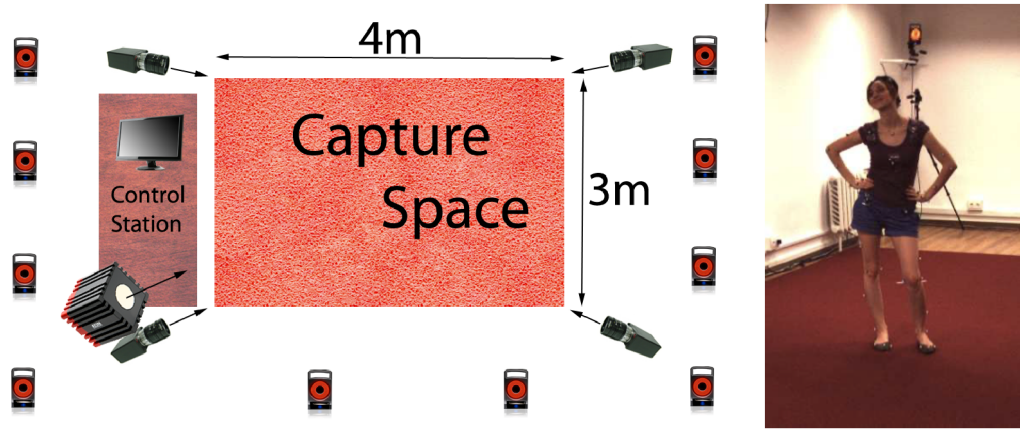
Carnegie Mellon University (CMU) dataset is a free dataset that contains human motion from 2,605 trials in 6 categories and 23 subcategories with frame dimension of 352×240 pixels and frame rate of 30fps. These 6 categories cover human motion such as human interactions, human interactions with environment, locomotive motions, physical activities & sports, situation & scenarios, and test motions. Fig. 3.1 shows the some samples from CMU dataset.

3.3 Human 3.6m Dataset

Human 3.6m, Human 3.6 is the current biggest dataset for human motion that contains 3.6 million 3D human poses and corresponding images. The dataset contains motion capture data which is ready to use for the human motion prediction. It consists of 15 activities including

(a) (b) (c)

Fig.3.1: Our dataset feature extraction process



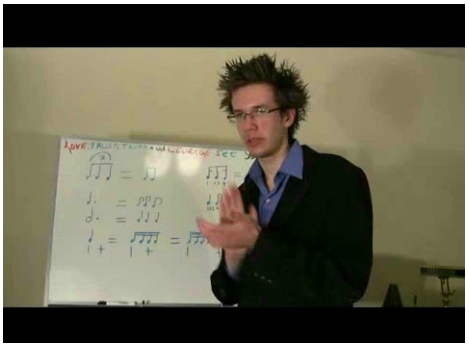
(a) Camera arrangement (b) Sample

Fig.3.2: Human 3.6m dataset

periodic activities like "walking" and non-periodic activities such as "discussion" and "taking photo" that were performed by seven different professional actors. Vicon motion capture system recorded the human 3.6m dataset which provides high quality 3D body joint locations in the global coordinate sampled at 50 frames per second (fps).

3.4 HMDB Dataset

HMDB dataset is a collected database from various sources, mostly from movies, and a small proportion from public databases such as the Prelinger archive, YouTube and Google videos. The dataset contains 6,849 clips divided into 51 action categories, each containing a minimum of 101 clips. The dataset is categorized into 6 action categories which are general facial actions, facial action with object manipulation, general body movements, body movement with interaction, and body movement for human interaction.



(a) Good quality

(b) Medium quality

Fig.3.3: HMDB dataset

3.5 JHMDB1 Dataset

JHMDB1 dataset is the extention of HMDB dataset with the joint-annotated samples.

Fig.3.4: JHMDB dataset

Chapter 4

Proposed Method

Fig. 4.1 shows a block diagram of the proposed method for human motion prediction system in this research. The system receives inputs from the video samples, these videos will later be processed to provide the feature of human body pose as a coordinate data divided by human body parts. We set goal to predict 1 second ahead of the motion, and we prepared 30 fps videos which include simple motions such as hand gesture and walking movement. Nodes are defined by human body parts which cover head, neck, shoulders, elbows, wrists, hip, knees, and ankles.

Before we proceed to obtain the prediction, we need to convert the coordinate data into the movement data which contains distance and direction. These movement data obtained from the change of the coordinate from the frame F_i to the frame F_{i+30} with Euclidean formula. After the movement data has been obtained, we proceed to the processing method that includes RNN-LSTM for the prediction.

4.1 Human Pose Feature Extraction

This section explains about the input for the system to collect the data from the features of the human body pose. We proposed 3 methods to obtained the suitable data as we need it in the processing to obtain the prediction results.

4.1.1 Human pose estimation

As the input of dataset have been prepared which is a video that contains a human feature with some motions inside the frame. We need to extract the data of human body pose by parts from the video. For this feature extraction we use OpenPose to obtain the pose estimation as

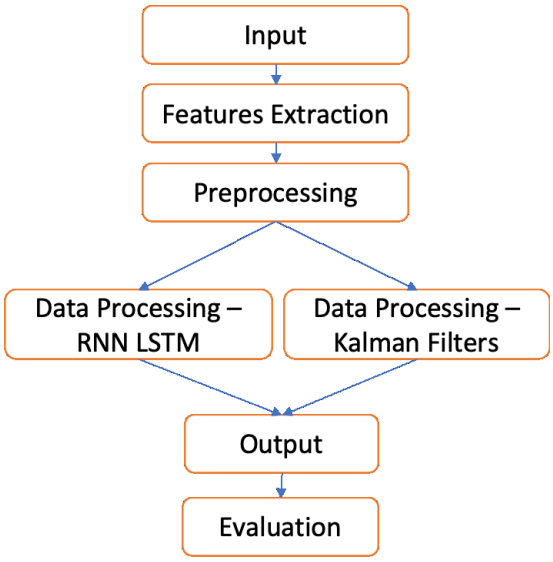


Fig.4.1: Block diagram of the proposed method

a coordinate values x, y of each body part. OpenPose with COCO dataset keypoints obtains the x and y coordinate value of 18 body points (nose, neck, left/right shoulders, left/right elbows, left/right wrists, left/right hips, left/right knees, and left/right ankles,left/right eyes, left/right ears).

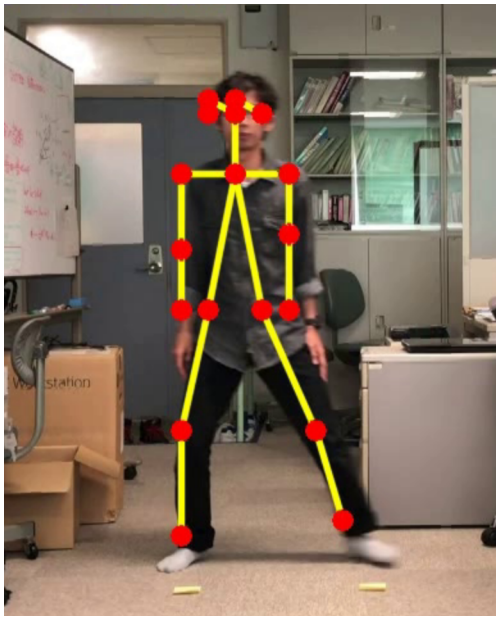


Fig.4.2: OpenPose result on our dataset

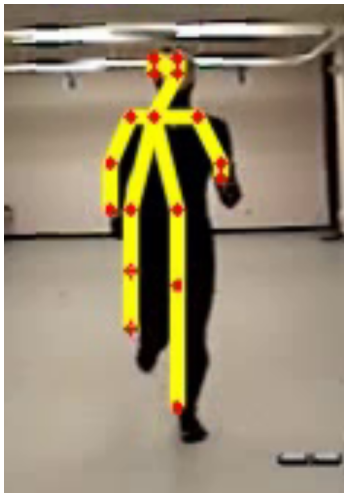


Fig.4.3: OpenPose result on CMU dataset : running

Fig.4.4: OpenPose result on CMU dataset : walking

4.1.2 Human recognition

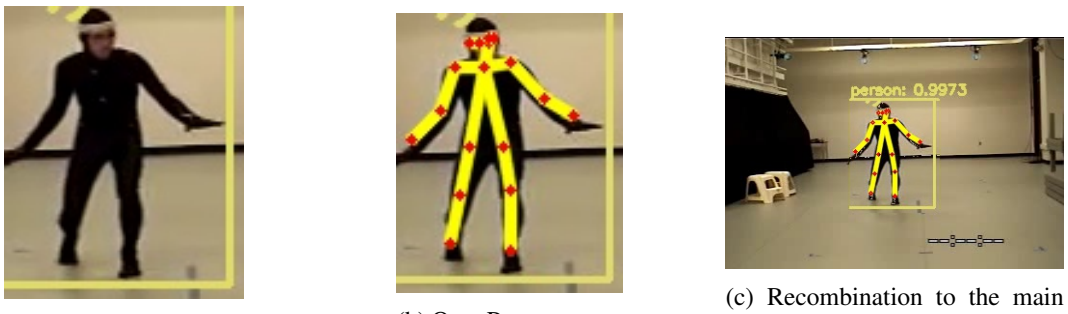
Fig. 4.1 shows the failed estimation that is obtained from OpenPose. This kind of error makes the data from OpenPose is not reliable to be used in the prediction process. Also, it makes the sequence data even more varied with a sudden motion, thus, it effects the prediction results. For this problem, we use YOLOv3 extracts human body bounding box crop the frame which only contains the human feature inside. Fig. 4.5a and Fig. 4.6a show the feature extraction process from implementation of YOLOv3, then the frame is processed in OpenPose as shown in Fig. 4.5b and Fig. 4.6b. After that we recombine the result from OpenPose to the main frame with respect to the position and size of the cropped frame to the main frame as shown in Fig. 4.5c and Fig. 4.6c.

(a) Cropped area by YOLOv3

(b) OpenPose pose estimation

(c) Recombination to the main frame

Fig.4.5: Our dataset feature extraction process



(a) Cropped area by YOLOv3

(b) OpenPose pose estimation

(c) Recombination to the main frame

Fig.4.6: CMU dataset feature extraction process

4.1.3 Data preprocessing

The obtained coordinate data x, y from the previous process are containing the absolute coordinate value in the processed image (i.e. $[438.2608695652174, 176.08695652173913]$). This type of data will give another burden to RNN-LSTM to memorize all of the specific number. As well as we need to prepare the timestep as the goal we demand to be obtained is 1 second in a video with 30 fps (frame per second) property. It means the timestep should be 30 frame consecutively in a form of sequence data as shown in Fig. 1.1.

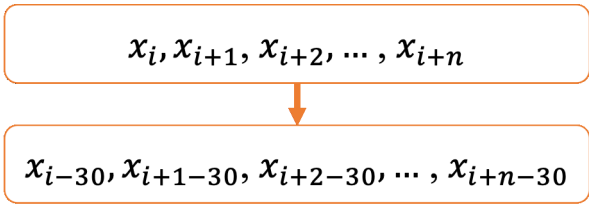


Fig.4.7: Sequence data with change of 30 frame

Next, we compare these 2 sequence data in order to obtain the distance d and direction θ

data by :

$$d_i = \sqrt{(x_i - x_{i-fs})^2 + (y_i - y_{i-fs})^2} \tag{4.1}$$

$$\theta_i = \arcsin(y_i - y_{i-fs}/d_i) \tag{4.2}$$

where fs is the frame step which has a constant value of 30 since the proposed method estimate the one second ahead motion and the input video has 30 fps property. From this process, we obtained the sequence data which contains distance d and direction θ. This type of data is also suitable for the next method.

4.2 Kalman Filter

Kalman Filter has two steps. The first step is predicting which it makes a first guess about what we think is true (an estimate) and how certain we are that is true (uncertainty). Next, Kalman Filter makes a new guess by using a weighted average. More certain numbers are more important in this weighted average. After doing these two steps, we use the new guess to start these steps again. In Kalman Filter, we have two important function to predict the sequence data. At i-th frame, the proposed method predicts the distance d_i and direction θ_i by using Eq. 4.3 and Eq. 4.4, respectively :

$$d_i = d_{i-1} + \left(\frac{(\sigma_i \cdot d_{i-1} + \sigma_c \cdot \delta_i)}{\sigma_i \cdot \sigma_c} \right) \tag{4.3}$$

$$\theta_i = \theta_{i-1} + \left(\frac{(\sigma_i \cdot \theta_{i-1} + \sigma_c \cdot \delta_i)}{\sigma_i \cdot \sigma_c} \right) \tag{4.4}$$

where d_i and θ_i respectively represent distance value and direction value on frame i-th, σ_i is the initial cost at first and an updated value of the d_i, σ_c represents the contant value of noise, δ_i represents the measurement data which produced by the feature extraction method.

4.3 RNN-LSTM

LSTM is an extended version of RNN which has the extended memory to memorize not only the short term of the sequence data, but further long term of the sequence data. LSTM networks were discovered by Hochreiter and Schmidhuber in 1997 [5]. LSTM works even given long delays between significant events and can handle signals that mix low and high frequency components. The Kalman Filter based pose estimation may fail when the human moves suddenly. This study proposes the RNN-LSTM based human motion estimation

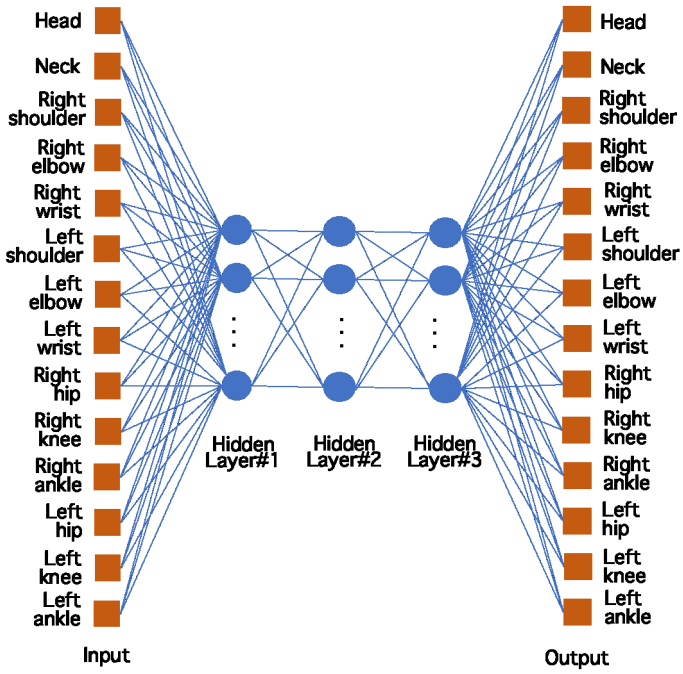


Fig.4.8: Neural network model

method. In this research, the input will be 14 nodes of human body parts and we use three stacked hidden layers for the learning model in RNN-LSTM. Fig. 4.8 shows the illustration of proposed neural networks model for RNN-LSTM. The last output will be 14 nodes as well as the input. Some other related researches [1, 7, 8] used three stacked layer RNN-LSTM as well.

We used the Mean Squared Error (MSE) that is defined by the following equation to estimate the loss value in the training of RNN-LSTM.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{x} - x_i)^2 \tag{4.5}$$

to estimate the loss value in the training process.

Chapter 5

Experimental Evaluation

This section explains about how do we perform the proposed method for this research, and evaluate it to measure the performance of the proposed method.

5.1 Dataset

As the input for the conducted system in this research, the dataset must be fit with the theme of this research. Since this study is about human motion prediction by RGB camera, we need to prepare the video samples which contain the feature of human with full body inside the frame, the sample human generate the typical motions of the human. We prepared 2 sample videos as shown in Fig. 5.1 which contains the simple motions such as hand gesture, moving aside, hand gesture + moving aside, and sitting motion. However, not only the dataset from us, there are many open dataset that contains human motion. Carnegie Mellon University (CMU) provides the open dataset for human motion with 2,605 trials on frame dimension of 352×240.

(a) Sample 1

(b) Sample 2

(c) Sample 3

Fig.5.1: Samples from our dataset which contains simple human motions such as hand gesture, moving aside, and sitting motion.

(a) Running straight (b) Running straight turn right (c) Walking stealthily

Fig.5.2: Sample from CMU dataset which contains more complex human motion such as running straight, running straight then turning right, and walking stealthily.

5.2 Training for prediction

From the previous chapter, the proposed method for this research has been explained. We are using two method for the sequence data prediction. The first one is RNN-LSTM and the second one is Kalman Filter. The input of the network are the sequence data of human body parts which includes 14 nodes obtained by feature extraction. We set the RNN-LSTM with 3 stacked hidden layers to optimized the prediction. Then as well as the input, output of the RNN-LSTM will have the same sequence data structure of 14 nodes which contains movement data array distance d_i and θ_i. As detail of RNN-LSTM, we set the model with 5000 epochs, the batch size was 128, and framestep was 30 frames.

For the prediction by Kalman Filter, the sequence data are compiled node by node. We applied the algorithm to predict the current position by calculating the previous node as describe in Chapter 3.2.

5.3 Evaluation

As the comparison of the prediction result that have been done by RNN-LSTM and Kalman Filter. We evaluate the distance from ground truth coordinate to the prediction coordinate. The ground truth coordinate is the actual coordinate that is obtained from the feature extraction in the $i + 30$ th frame f_{i+30}. While the prediction coordinate is the position of the node that are obtained from the prediction method and converted to coordinate data by the following equation.

$$x_p = x_i + (sin(\theta_i) \times d_i) \tag{5.1}$$

$$y_p = y_i + (cos(\theta_i) \times d_i) \tag{5.2}$$

where x_p represents the predicted coordinate x, and y_p represents the predicted coordinate y, x_i represents the current coordinate x, as well as y_i represents the current coordinate$x$, θ_i is the direction value on $i-$th from the prediction result, and d_i is the distance value on $i-$th from the prediction result.

Then, the prediction coordinates by nodes were ready to be shown in the video. We calculate the prediction evaluation distance to know how far is the prediction result by nodes to the ground truth by the following equation.

$$E = \sqrt{(x_{i+fs} - x_p)^2 + (y_{i+fs} - y_p)^2} \tag{5.3}$$

where E represents the evaluation distance, fs is the framestep which is defined as a contant value of 30 in this research.

We calculate the frequency of the prediction results that the distance are less than the limited value which is 1.8% of the diagonal frame size of the video. This limitation value of evaluation is also used for CMU dataset to calculate the frequency of the good result based on the distance.

5.4 Results and Discussion

In this chapter, we show the result of the prediction as well as its evaluation results to evaluate the performance of our proposed method. We separate the result and its evaluation by the dataset to compare RNN-LSTM prediciton result and Kalman Filter prediction result.

5.4.1 Prediction result using RNN-LSTM

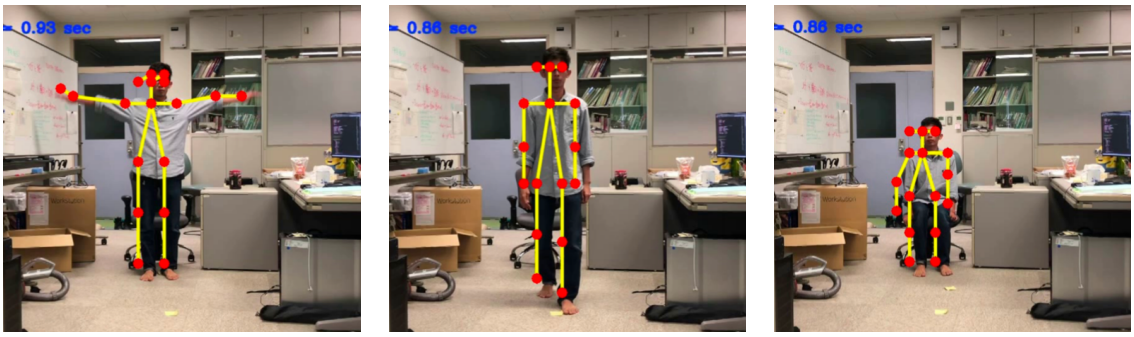
Our dataset - Experiment sample 1

Fig. 5.3, Fig. 5.4, and Fig. 5.5 show the implementation process in the video from the proposed method. As shown in Fig. 5.5, the red points are the current position of the node, while the blue points are the node. For the experiment sample 1 we input the node one by one to the model, and compile it by each nodes. One node will be processed by one learning process. With this architecture, RNN-LSTM could store more memory for the learning process. However, It does not consider the related nodes each other. So, the results stand by itself independently.

Fig. 5.6 shows the percentage of succesful results based on the distance value that are lower than 1.8% of the diagonal frame size of the video which is 20 pixels in our dataset. The results are mostly satisfied enough accuracy with around 80% of the results are close to the ground

(a) Hand gesture motion (b) Walk forward motion (c) Sitting motion

Fig.5.3: Our dataset : sample 1



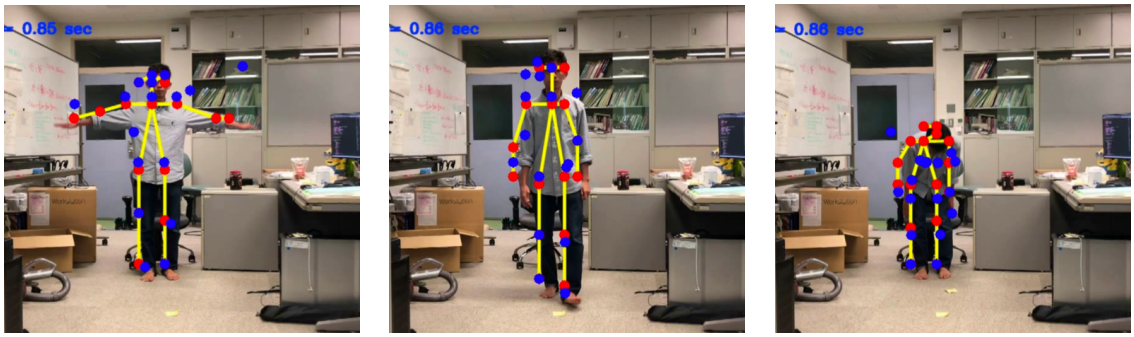
(a) Hand gesture motion (b) Walk forward motion (c) Sitting motion

Fig.5.4: Feature Extraction from our dataset : sample 1

truth. However, the elbow and wrist nodes are not as satisfiable as the other nodes. Based on the motions in the video, elbows and wrists move more than other nodes. The prediction result on elbow and wrist nodes only obtained around 30% to 50% which means most of the prediction data are away from the ground truth.

Our dataset - Experiment sample 2

Fig. 5.7, Fig. 5.8, and Fig. 5.9 show the implementation process in the video from the proposed method with comparison of Kalman Filter on Fig. 5.7, Fig. 5.8, and Fig. 5.9. While Fig. 5.9c shows no blue points, because of the RNN-LSTM could not predict since it has its limitation on memorizing all of the sequence data by the hardware. From this experiment, we evaluate the result by percentage of the evaluation distance value that is lower than 1.8% of the diagonal frame size, mean, median and standard deviation value. As shown in Fig. 5.10, RNN-LSTM prediction results are varied, 47% of the result were succesful based on the



(a) Hand gesture motion (b) Walk forward motion (c) Sitting motion

Fig.5.5: Prediction Result from our dataset : sample 1

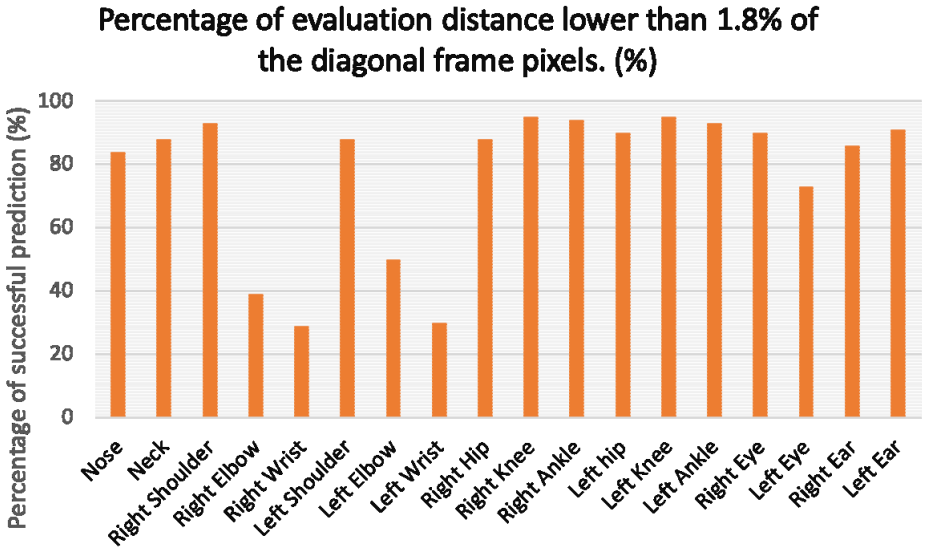
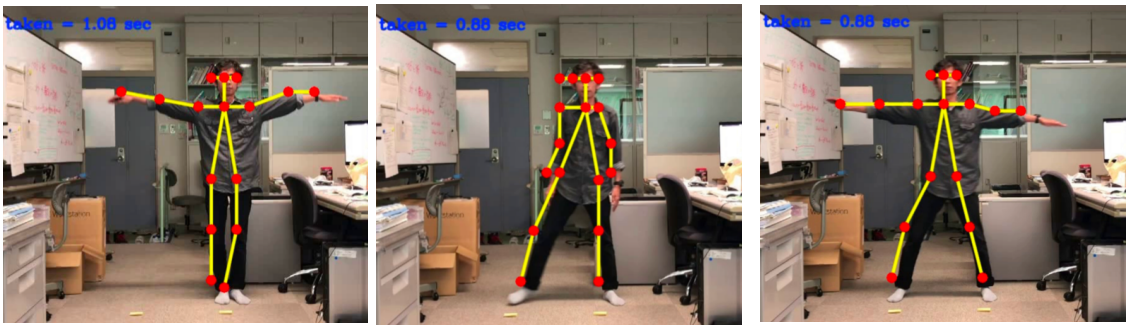


Fig.5.6: Percentage of the evaluation distance obtained from RNN-LSTM prediction result by each node lower than 1.8% of the diagonal frame size.

percentage of the evaluation distance value that is lower than 1.8% of the diagonal frame size. While, the average of the prediction result are varied from 23 pixels to 59 pixels away from the ground truth. We can see the variation of the prediction result on the median value, nose shows most of the data are close to the ground truth with median value 0.0058 pixel, the same goes to neck, right shoulder, right knee, left hip, left ankle, right eye, left eye, and left ear with the max median value is 4.66 pixels. However, the other nodes show different results range from 22.97 pixels to 49.5 pixels. Standard deviation shows the correlation between the value in the evaluation distance while the range of the value are from 31 to 50 which means the evaluation distances do not correlate a lot each other. The detail can be seen on Table 5.1

(a) Hand gesture (b) Move to the right (c) Hand gesture + move to the right

Fig.5.7: Our dataset : sample 2



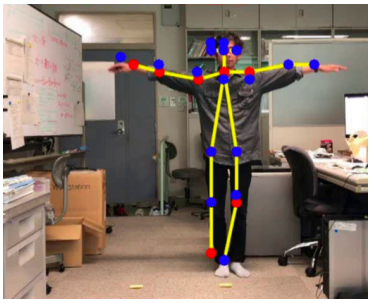
(a) Hand gesture (b) Move to the right (c) Hand gesture + move to the right

Fig.5.8: Feature Extraction from our dataset : sample 2

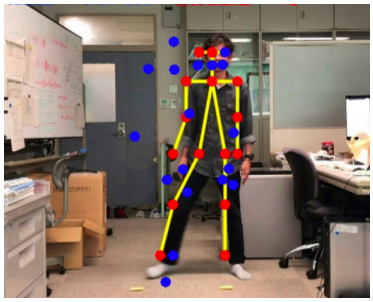
CMU dataset

Fig. 5.11 and Fig. 5.12 show the prediction result on video respectively. The red points are the current positions and the blue points are the prediction positions. Fig. 5.13 shows the percentage of the succesful prediction on CMU dataset : walking stealthily and running.

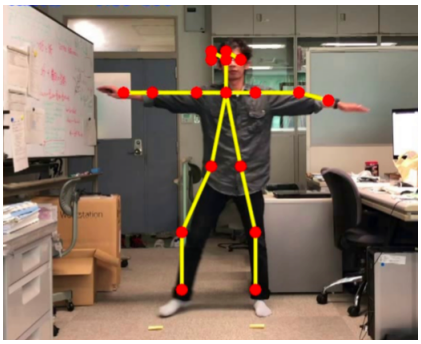
Based on its evaluation, most of the nodes obtained 40% of the succesful predictions. However, for the node of right shoulder, right eye, left eye, right ear, left ear obtained around or less than 20% of the prediction were succesful. These motions from CMU dataset are basically more complex than our dataset.



(a) Hand gesture



(b) Move to the right



(c) Hand gesture + move to the right

Fig.5.9: Prediction result by RNN-LSTM on our dataset : sample 2

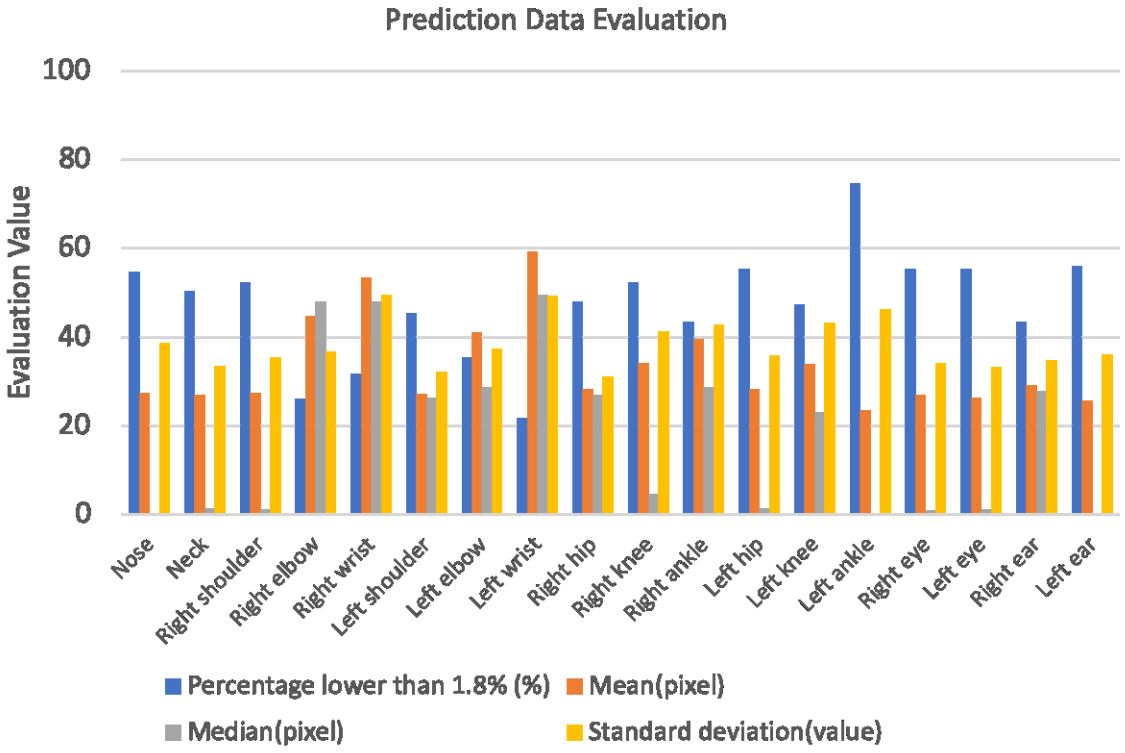
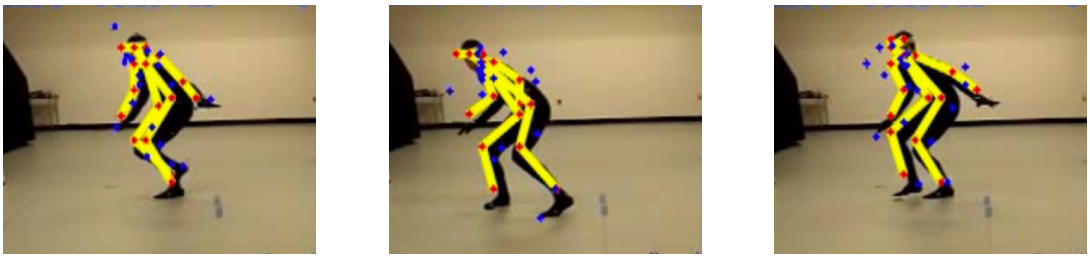


Fig.5.10: Evaluation of predicted results obtained from RNN-LSTM prediction result by each node based on the percentage of the value lower than 1.8% of the diagonal frame size, mean, median, and the standard deviation.

Table.5.1: Evaluation of predicted results obtained from RNN-LSTM prediction result by each node based on the percentage of the value lower than 1.8% of the diagonal frame size, mean, median, and the standard deviation.

Nodes	Percentage(%)	Mean(pixel)	Median(pixel)	Std.(value)
Nose	54.658	27.422	0.006	38.645
Neck	50.310	26.829	1.229	33.466
Right Shoulder	52.174	27.280	1.056	35.258
Right Elbow	26.087	44.742	47.855	36.673
Right Wrist	31.677	53.380	47.984	49.533
Left Shoulder	45.342	27.209	26.286	32.137
Left Elbow	35.400	40.939	28.515	37.196
Left Wrist	21.739	59.234	49.494	49.279
Right Hip	47.826	28.134	26.975	31.097
Right Knee	52.174	34.087	4.659	41.118
Right Ankle	43.478	39.434	28.523	42.680
Left Hip	55.279	28.185	1.279	35.841
Left Knee	47.205	33.769	22.970	43.252
Left Ankle	74.534	23.498	0.225	46.187
Right Eye	55.279	26.921	0.914	34.032
Left Eye	55.279	26.211	1.080	33.229
Right Ear	43.478	29.087	27.650	34.591
Left Ear	55.900	25.662	0.004	36.051



(a) (b) (c)

Fig.5.11: Prediction result by RNN-LSTM on CMU dataset : walking steathily

(a) (b) (c)

Fig.5.12: Prediction result by RNN-LSTM on CMU dataset : running

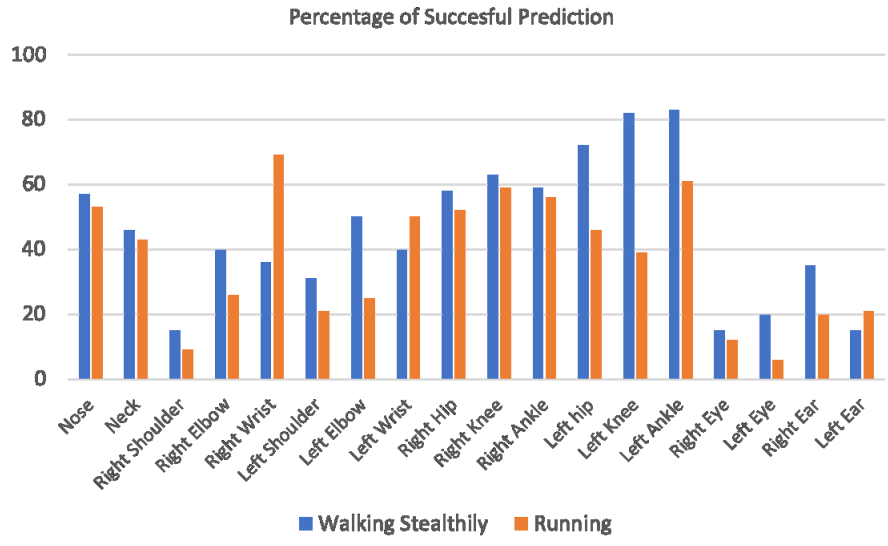
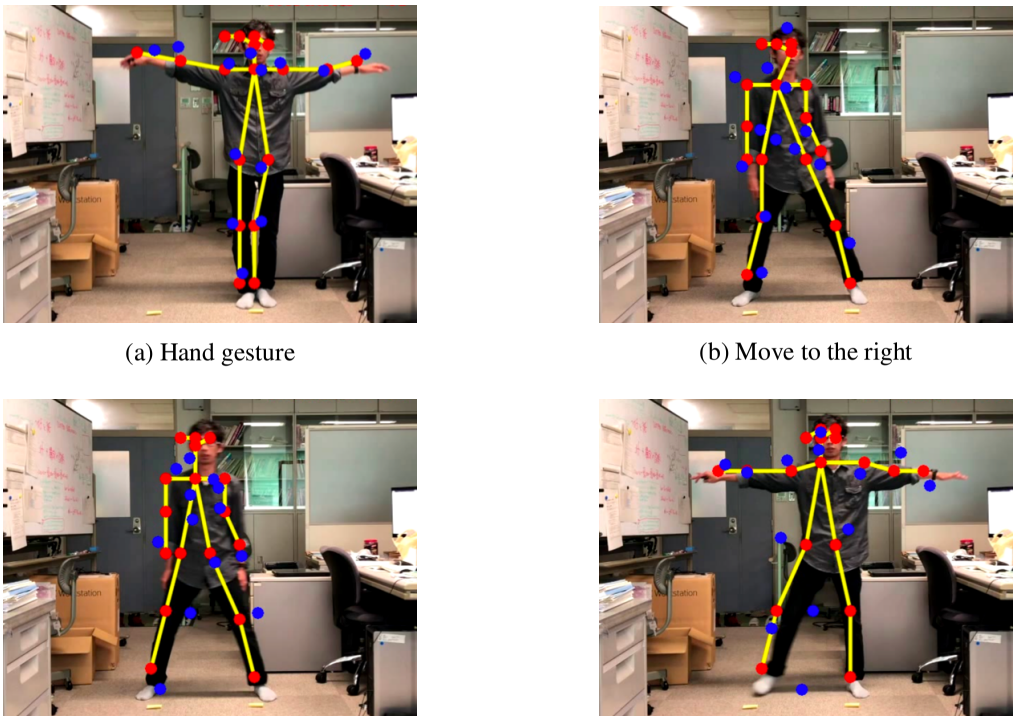


Fig.5.13: Evaluation of predicted results obtained from RNN-LSTM prediction result by each node and motions based succesful percentage.

5.4.2 Prediction result using Kalman Filter

Our dataset

Fig. 5.14 shows the results of the Kalman Filter prediction on our dataset : sample 2. Kalman Filter use the statistical calculation based on the previous state to predict the current state. The prediction does not need to memorize the sequence data like RNN-LSTM, thus, Kalman Filter has no limitation to predict even the sequence data is huge. In this experiment, Kalman Filter can predict the human motion in the video, we decide to conclude the nose node as the head node and remove the eyes and ears node since it will not effect much to the motion. The evaluation result by the succesful percentage can be seen on Fig. 5.15, where most of the results are satisfiable. However, Moving to the left motion on left right ankle, left knee, and left ankle predictions were just 60% that set as the succesful prediction which is the prediction with evaluation distance are lower than 1.8% of the diagonal frame size. Fig. 5.16 shows the evaluation distance statistics defined by mean, median, and standard deviation value by each node.



(a) Hand gesture

(b) Move to the right

(c) Move to the left

(d) Hand gesture + move to the right

Fig.5.14: Prediction result by Kalman Filter from our dataset : sample 2

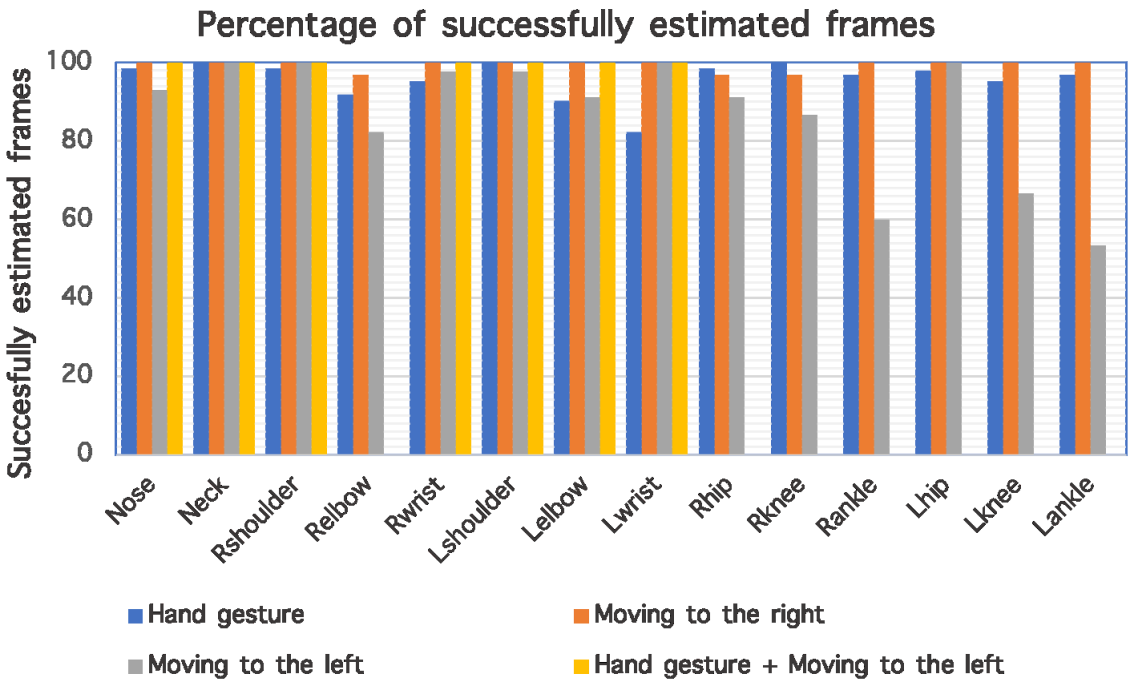


Fig.5.15: Evaluation of predicted results obtained from Kalman Filter prediction result by each node and motions based on the percentage of the value lower than 1.8%.

CMU dataset

Fig. 5.17 shows the prediction result that is done by Kalman Filter. While Fig. 5.18 shows the evaluation based on the percentage of the succesful prediction. Most of the nodes obtained around 70% to 80% of the prediction were succesful. Even though, similar to the result of RNN-LSTM on our dataset sample 1, elbows and wrists show lower percentage of the prediction than the other nodes. More detail about the result, Fig. 5.19 shows the statistical evaluation by mean, median and standard deviation value. On average, the prediction results are around 5 pixels away to the ground truth and median value with 3.8 pixels. Standard deviation value with 30 shows that the prediction result are far correlated to each other.

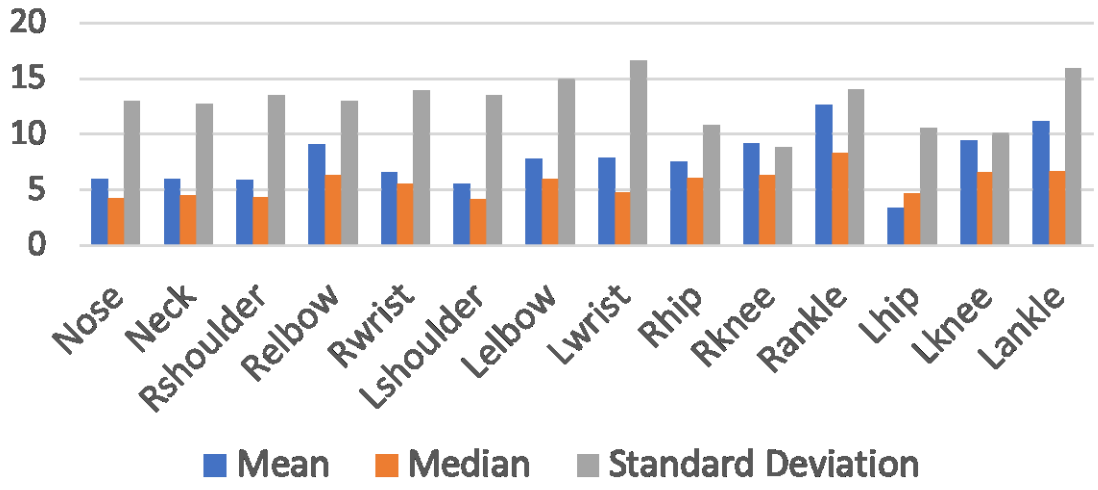


Fig.5.16: Evaluation of predicted results obtained from Kalman Filter prediction result by each node and motions based on mean, median, and standard deviation value.

(a) (b) (c)

Fig.5.17: Prediction result by RNN-LSTM on CMU dataset : running

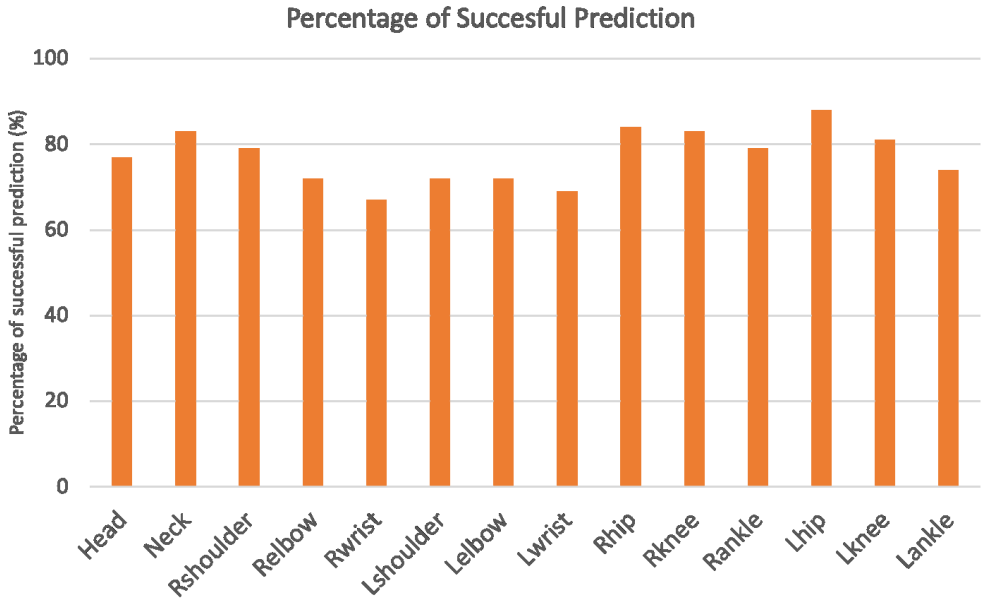


Fig.5.18: Evaluation of predicted results obtained from Kalman Filter prediction result by each node and motions based on succesful percentage.

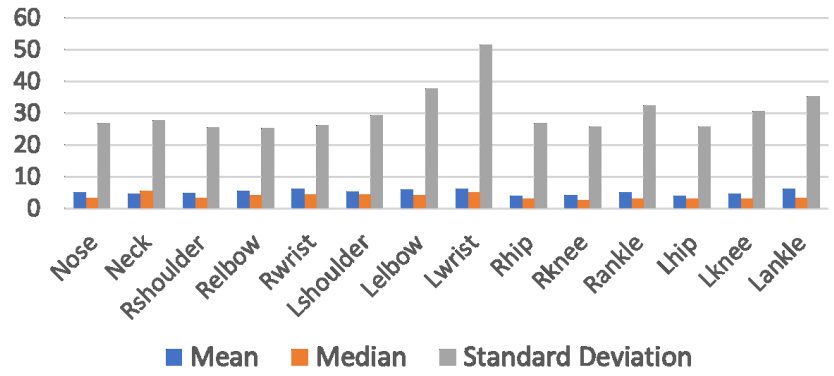


Fig.5.19: Evaluation of predicted results obtained from Kalman Filter prediction result by each node and motions based on mean, median, and standard deviation value.

Chapter 6

Summary

This chapter explain about this research conclusion from all experiment and the future works for the future development.

6.1 Conclusion

Based on the result of all experiment, we have proposed the human movement prediction with RNN-LSTM and Kalman Filter based on RGB camera for the motion prediction of one second ahead. We used samples of video that cover hand gesture motion, sideways moving, sitting motion, walking, running. As the results, Kalman Filter reached 93% in average, and RNN-LSTM reached 75% in average on our dataset. While Kalman Filter reached 77% in average, and RNN-LSTM reached 52% in average on CMU dataset. Mostly, Kalman Filter show better accuracy then RNN-LSTM and based on the human motions, motion such as hand gesture and moving to the right side are easier than more complex motion like hand gesture and moving to the left side. Even though, The result showed most of the predictions are close to the correct position that is a prediction for one second of human movement. We confirmed the validity of the RGB based method with the unstable data in the simple human motion case from the result, and we conclude that this is an important step to realize the prediction of more complex human motion.

6.2 Future works

For the future development, we need to realize the combination of RNN-LSTM and Kalman Filter for the human motion prediction. For the comparison of our method, we should compare with another research performance with the same focus on human motion

prediction. More experiment is needed for this research to advance the more complex motion.

Appendix A

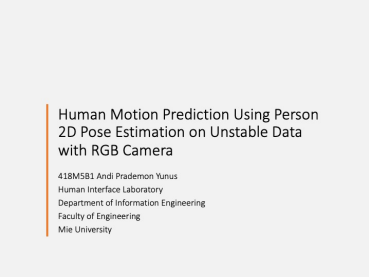
A.1 Program file location

All of this research's program locate in the following directory in the human interface laboratory's server.

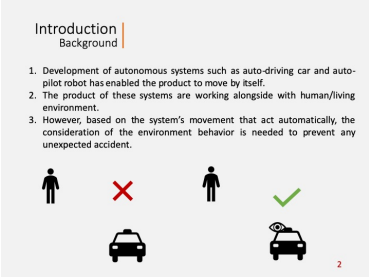
1. /net/xserve0/users/andi/

A.2 Thesis defense presentation

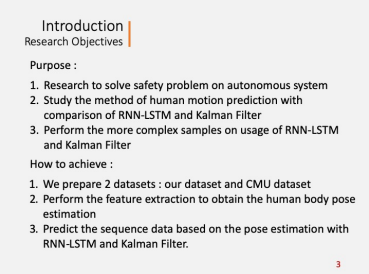
Thesis defense was presented by the following presentation :



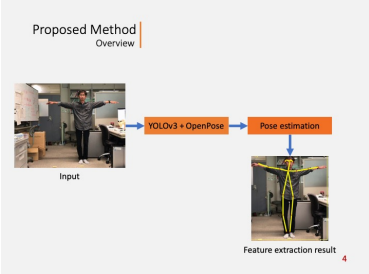
(a)



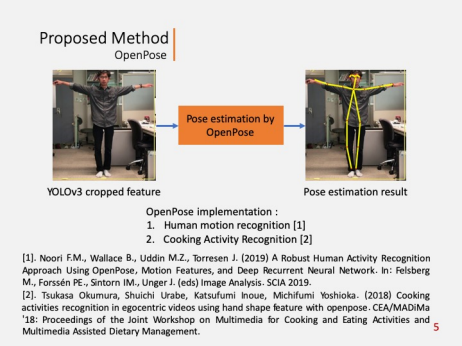
(b)



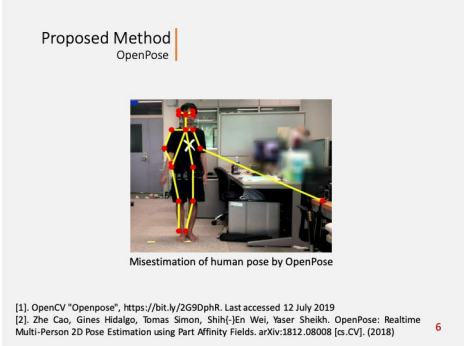
(c)



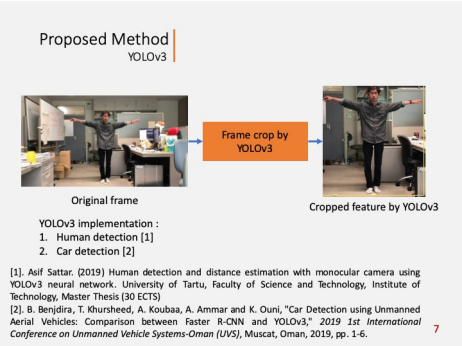
(d)


Proposed Method
OpenPose
Pose estimation by OpenPose
YOLOv3 cropped feature
Pose estimation result
OpenPose implementation :
1. Human motion recognition [1]
2. Cooking Activity Recognition [2]
[1]. Noori F.M., Wallace B., Uddin M.Z., Torresen J. (2019) A Robust Human Activity Recognition Approach Using OpenPose, Motion Features, and Deep Recurrent Neural Network. In: Felsberg M., Forssén PE., Sintorn IM., Unger J. (eds) Image Analysis. SCIA 2019.
[2]. Tsukasa Okumura, Shuichi Urabe, Katsufumi Inoue, Michifumi Yoshioka. (2018) Cooking activities recognition in egocentric videos using hand shape feature with openpose. CEA/MADiMa '18: Proceedings of the Joint Workshop on Multimedia for Cooking and Eating Activities and Multimedia Assisted Dietary Management.
5

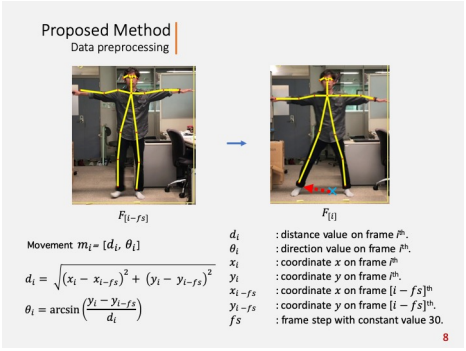
(a)


Proposed Method
OpenPose
Misestimation of human pose by OpenPose
[1]. OpenCV "Openpose", https://bit.ly/2G9DphR. Last accessed 12 July 2019
[2]. Zhe Cao, Gines Hidalgo, Tomas Simon, Shih(-)En Wei, Yaser Sheikh. OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. arXiv:1812.08008 [cs.CV]. (2018)
6

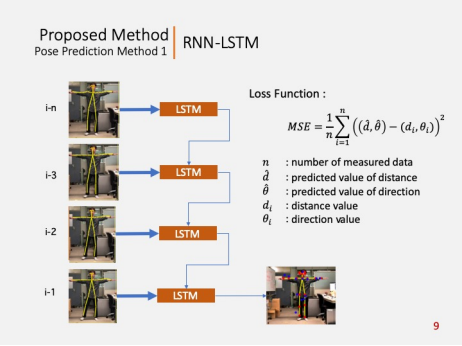
(b)


Proposed Method
YOLOv3
Frame crop by YOLOv3
Original frame
Cropped feature by YOLOv3
YOLOv3 implementation :
1. Human detection [1]
2. Car detection [2]
[1]. Asif Sattar. (2019) Human detection and distance estimation with monocular camera using YOLOv3 neural network. University of Tartu, Faculty of Science and Technology, Institute of Technology, Master Thesis (30 ECTS)
[2]. B. Benjdira, T. Khursheed, A. Koubaa, A. Ammar and K. Ouni, "Car Detection using Unmanned Aerial Vehicles: Comparison between Faster R-CNN and YOLOv3," 2019 1st International Conference on Unmanned Vehicle Systems-Oman (UVS), Muscat, Oman, 2019, pp. 1-6.
7

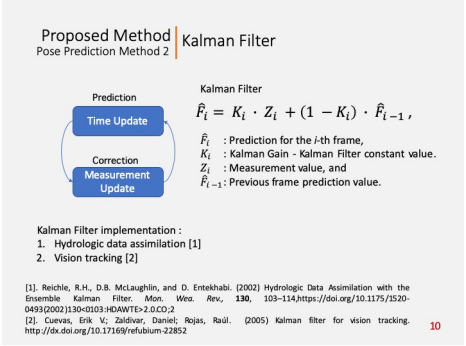
(c)


Proposed Method
Data preprocessing
$F_{[i-fs]}$
$F_{[i]}$
Movement $m_i = [d_i, \theta_i]$
$d_i = \sqrt{(x_i - x_{i-fs})^2 + (y_i - y_{i-fs})^2}$
$\theta_i = \arcsin\left(\frac{y_i - y_{i-fs}}{d_i}\right)$
d_i : distance value on frame i^{th}.
θ_i : direction value on frame i^{th}.
x_i : coordinate x on frame i^{th}
y_i : coordinate y on frame i^{th}.
x_{i-fs} : coordinate x on frame $[i-fs]^{th}$
y_{i-fs} : coordinate y on frame $[i-fs]^{th}$.
fs : frame step with constant value 30.
8

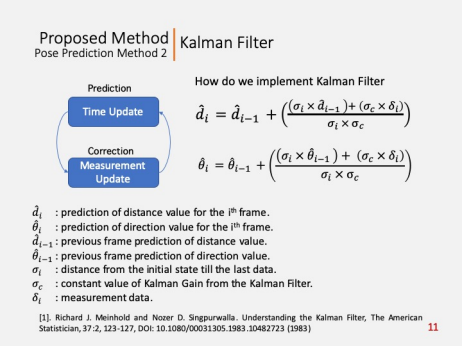
(d)


Proposed Method
Pose Prediction Method 1
RNN-LSTM
i-n
i-3
i-2
i-1
LSTM
LSTM
LSTM
LSTM
Loss Function :
$MSE = \frac{1}{n}\sum_{i=1}^{n}\left((\hat{d}, \hat{\theta}) - (d_i, \theta_i)\right)^2$
n : number of measured data
$\hat{d}$: predicted value of distance
$\hat{\theta}$: predicted value of direction
d_i : distance value
θ_i : direction value
9

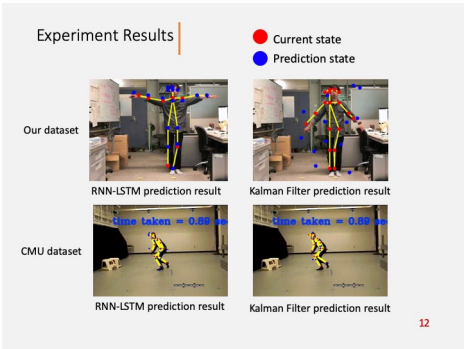
(e)


Proposed Method
Pose Prediction Method 2
Kalman Filter
Prediction
Time Update
Correction
Measurement Update
Kalman Filter
$\hat{F}_i = K_i \cdot Z_i + (1 - K_i) \cdot \hat{F}_{i-1}$,
$\hat{F}_i$: Prediction for the i-th frame,
K_i : Kalman Gain - Kalman Filter constant value.
Z_i : Measurement value, and
$\hat{F}_{i-1}$: Previous frame prediction value.
Kalman Filter implementation :
1. Hydrologic data assimilation [1]
2. Vision tracking [2]
[1]. Reichle, R.H., D.B. McLaughlin, and D. Entekhabi. (2002) Hydrologic Data Assimilation with the Ensemble Kalman Filter. Mon. Wea. Rev., 130, 103–114,https://doi.org/10.1175/1520-0493(2002)130<0103:HDAWTE>2.0.CO;2
[2]. Cuevas, Erik V; Zaldívar, Daniel; Rojas, Raúl. (2005) Kalman filter for vision tracking. http://dx.doi.org/10.17169/refubium-22852
10

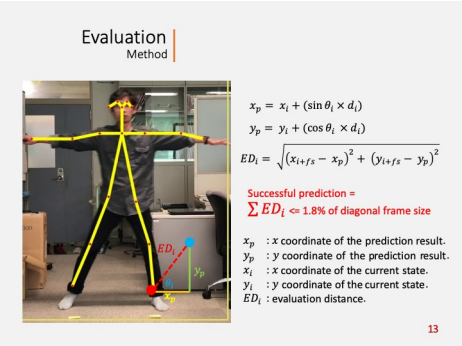
(f)


Proposed Method
Pose Prediction Method 2
Kalman Filter
Prediction
Time Update
Correction
Measurement Update
How do we implement Kalman Filter
$\hat{d}_i = \hat{d}_{i-1} + \left(\frac{(\sigma_i \times \hat{d}_{i-1}) + (\sigma_c \times \delta_i)}{\sigma_i \times \sigma_c}\right)$
$\hat{\theta}_i = \hat{\theta}_{i-1} + \left(\frac{(\sigma_i \times \hat{\theta}_{i-1}) + (\sigma_c \times \delta_i)}{\sigma_i \times \sigma_c}\right)$
$\hat{d}_i$: prediction of distance value for the i^th frame.
$\hat{\theta}_i$: prediction of direction value for the i^th frame.
$\hat{d}_{i-1}$: previous frame prediction of distance value.
$\hat{\theta}_{i-1}$: previous frame prediction of direction value.
σ_i : distance from the initial state till the last data.
σ_c : constant value of Kalman Gain from the Kalman Filter.
δ_i : measurement data.
[1]. Richard J. Meinhold and Nozer D. Singpurwalla. Understanding the Kalman Filter, The American Statistician, 37:2, 123-127, DOI: 10.1080/00031305.1983.10482723 (1983)
11

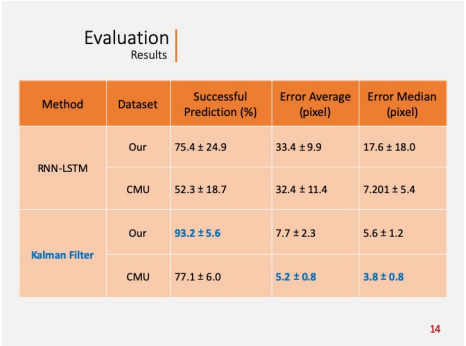
(a)


Experiment Results
Current state
Prediction state
Our dataset
RNN-LSTM prediction result
Kalman Filter prediction result
CMU dataset
Time taken = 0.89 ms
RNN-LSTM prediction result
Kalman Filter prediction result
12

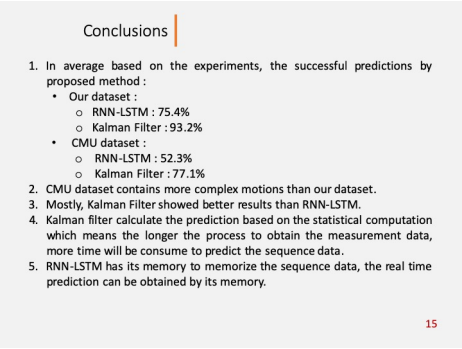
(b)


Evaluation
Method
$x_p = x_i + (\sin\theta_i \times d_i)$
$y_p = y_i + (\cos\theta_i \times d_i)$
$ED_i = \sqrt{(x_{i+fs} - x_p)^2 + (y_{i+fs} - y_p)^2}$
Successful prediction =
$\sum ED_i$ <= 1.8% of diagonal frame size
x_p : x coordinate of the prediction result.
y_p : y coordinate of the prediction result.
x_i : x coordinate of the current state.
y_i : y coordinate of the current state.
ED_i : evaluation distance.
13

(c)


Evaluation
Results
Method
Dataset
Successful Prediction (%)
Error Average (pixel)
Error Median (pixel)
RNN-LSTM
Our
75.4 ± 24.9
33.4 ± 9.9
17.6 ± 18.0
CMU
52.3 ± 18.7
32.4 ± 11.4
7.201 ± 5.4
Kalman Filter
Our
93.2 ± 5.6
7.7 ± 2.3
5.6 ± 1.2
CMU
77.1 ± 6.0
5.2 ± 0.8
3.8 ± 0.8
14

(d)


Conclusions
1. In average based on the experiments, the successful predictions by proposed method :
• Our dataset :
o RNN-LSTM : 75.4%
o Kalman Filter : 93.2%
• CMU dataset :
o RNN-LSTM : 52.3%
o Kalman Filter : 77.1%
2. CMU dataset contains more complex motions than our dataset.
3. Mostly, Kalman Filter showed better results than RNN-LSTM.
4. Kalman filter calculate the prediction based on the statistical computation which means the longer the process to obtain the measurement data, more time will be consume to predict the sequence data.
5. RNN-LSTM has its memory to memorize the sequence data, the real time prediction can be obtained by its memory.
15

(e)

Acknowledgement

I would first like to thank my thesis advisor Prof. Tetsushi Wakabayashi of Human Interface Laboratory at Mie University. The door to Prof. Wakabayashi was always open whenever I ran into a trouble spot or had a question about my research or writing. He consistently give me advice in my research and even the living in Japan. I am grateful to all Human Interface Laboratory members, for a wonderful experience that we gained together during my period of staying in Japan.

Afterward, I acknowledge the support of my teachers in Information Engineering Department of Mie University for the kindness and humble encouragement. As well as thanks to Japanese government for providing me with this scholarship to study at Mie University in Japan.

Special thanks to my beloved supervisor Prof. Naoki Isu for the warm kindness, wonderful and humble advice and recommendation for me to study in Mie University.

Finally, the completion of this thesis involved the support from my entire family and friends. I dedicated my work on this thesis for my beloved parents whom have sent me pray and support that are never stopped. Then, thanks to Indonesian Student Association for their full support and for enduring my long study trip away from Indonesia. Thank you very much for the love and support.

References

[1] M. Julieta, B. J. Michael, R. Javier, ”On Human Motion Prediction Using Recurrent Neural Networks,” arXiv:1705.02445v1, 2017

[2] Alex Graves, ”Generating Sequences With Recurrent Neural Networks,” arXiv:1308.0850v5 [cs.NE], 2014

[3] Che Zhengping, Purushotham Sanjay, Cho Kyunghyun, Sontag David, Liu Yan, ”Recurrent Neural Networks for Multivariate Time Series with Missing Values,” Scientific Reports 8:6085, 2018

[4] R. Akita, A. Yoshihara, T. Matsubara and K. Uehara, ”Deep learning for stock prediction using numerical and textual information,” IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS) pp. 1-6, 2016

[5] Hochreiter, Sepp, Schmidhuber, Jrgen, ”Long Short-term Memory Neural computation,” 9.1735-80.10.1162/neco.1997.9.8.1735. 1997

[6] Richard J. Meinhold and Nozer D. Singpurwalla, ”Understanding the Kalman Filter,” The American Statistician, 37:2, 123-127, 1983

[7] Tang Yongyi, Ma Lin, Liu Wei, Zheng Wei-Shi, ”Long-Term Human Motion Prediction by Modeling Motion Context and Enhancing Motion Dynamics,” In:935-941. 10.24963/ijcai.2018/130, 2018

[8] Wu, Erwin, Koike, Hideki, ”Real-time human motion forecasting using a RGB camera,” 1-2. 10.1145/3281505.3281598, 2018

[9] C. Yujiao,Z. Weiye, L. Changliu, T. Masayoshi, ”Human Motion Prediction using Adaptable Neural Networks” arXiv:1810.00781, 2018

[10] Katerina Fragkiadaki, Sergey Levine, Panna Felsen, Jitendra Malik, ”Recurrent Network Models for Human Dynamics,” arXiv:1508.00271 [cs.CV], 2015

[11] Ashesh Jain, Amir R. Zamir, Silvio Savarese, Ashutosh Saxena, ”Structural-RNN: Deep Learning on Spatio-Temporal Graphs,” arXiv:1511.05298 [cs.CV] CVPR, 2016

[12] S. Selvin, R. Vinayakumar, E. A. Gopalakrishnan, V. K. Menon and K. P. Soman,

”Stock price prediction using LSTM, RNN and CNN-sliding window model,” 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 1643-1647, 2017

[13] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang, ”Recurrent Neural Network for Text Classification with Multi-Task Learning,” Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-16), 2016

[14] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, Yaser Sheikh, ”OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields,” arXiv:1812.08008 [cs.CV], 2018

[15] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi, ”You Only Look Once: Unified, Real-Time Object Detection,” arXiv:1506.02640v5 [cs.CV], 2016

[16] OpenCV ”Openpose”, https://bit.ly/2G9DphR. Last accessed March 9th, 2020