Real-time Human Motion Detection And Classification

Farhan S. Khan, Salman A. Baset
GIK Institute, Toppi
u970031@giki.edu.pk, u970095@giki.edu.pk

Abstract

Tracking the movements of different body parts of human and classifying them in certain states is an important problem in human-computer interaction.

In this paper we propose a robust mechanism for the tracking and classification of movements of hands and legs of a person in real-time. The technique employs a localized and adaptive frame-based motion detection mechanism that not only caters for considerable intensity variations in the video streams, but also takes care of deformation in image, rotation in the human body and self-occlusion of body parts. To make system work in real-time, person wears different color bands on his/her hands and legs. The simulation built on this mechanism continuously processes the video stream, detects the movements of hands and legs and classifies them in certain states, which can be labeled as {left/right} hand punched, and {left/right} hand unpunched,{left/right} leg kicked, {left/right} leg unkicked.

Keywords: *Locality of Movements, Self-Occlusion, Range-based Mode Filter*

1. Introduction

In some computer games, a human-controlled (HC) sprite fights with a computer sprite. The actions of a sprite in such a game are punches, kicks and blocks and human sprite is usually controlled through a keyboard / joystick.

Consider a game setup where human-controlled sprite is not controlled through the keyboard. Rather the human actually kicks, punches and blocks and HC sprite emulates his actions—a kind of virtual fighting environment. Thus the human behaves as if he were in a fight. His actions are 'detected', 'classified' and 'replicated' on the HC sprite.

An Image Processing / Graphical system that implements such applications requires robust algorithms for motion detection and classification. It also requires some sensors to extract and track the movements of left and right hands / legs in a frame and to differentiate between left and right hand / leg. In the absence of any sensors, it is computationally very expensive to perform above-mentioned tasks. Thus the person wears different color bands (acting as sensors) on hands and legs so as to enable the system to swiftly track their movements and differentiate between identical body parts.

The system maintains the previous movements of hands and legs in a System History. Range-based Mode (RM) filter is used to extract moving parts from video frames. Self-Occlusion is checked upon unsuccessful object detection. Finally a classifier heuristic is used to classify each movement as a punch / unpunch, kick / unkick.

2. Related Work

Considerable work done has been done in the field of motion detection and segmentation. The target application in most of the cases [1,3,4] has been a video conferencing application and main focus remained on the segmentation of moving object as a whole and not the individual parts. There are few algorithms that focus on the motion detection and classification [2], which use a neural net mechanism to detect and track the movements of a person. But such algorithms are computationally expensive and unable to give real-time performance.

The main objective of this mechanism is to track the movements of different body parts separately, not the movement of body as a whole. In addition, it also classifies the movements in *real-time*.

3. System Architecture

We have developed an agent-based architecture, where independent modules are cascaded to form a pipeline.

The system consists of a camera-interaction module, five processing agents, and a database of previous movements as shown in Figure 1.

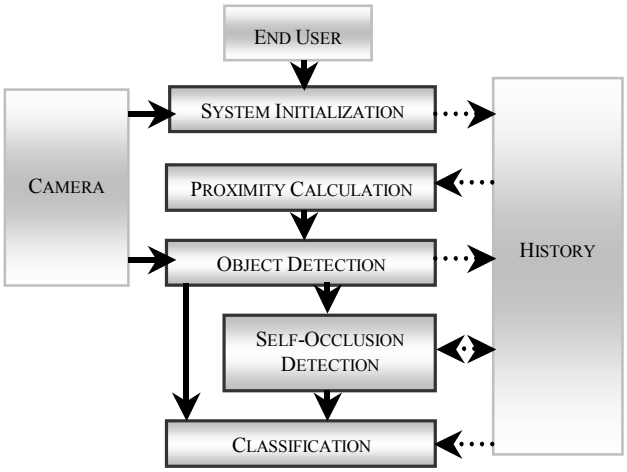

Figure 1: Agents based architecture

The end-user performs one time (in the beginning) system initialization using System Initialization Agent (SIA). SIA initializes History with the location of hands / legs and their respective band colors. Proximity calculation agent extracts the previous location of an object from history and using the principle of locality predicts a region in which the moving part is most likely to be found in the next frame. Using Range Mode(RM) Filter, Object Detection agent searches the body part in the region provided by the proximity calculation agent. If object detection is unsuccessful, then Self-Occlusion Detection Agent(SODA) checks whether self-occlusion has occurred or not, and in either case adopts a remedial strategy. It uses history of previous movements and a self-occlusion algorithm to find direction vector for the occluded part. This direction vector is used to predict the next move and position of that part. Finally classification agent classifies the movements of body parts in certain defined states.

3.1. Locality of Movement (LoM)

Movement of hands and legs with respect to the whole body are confined to a limited region, i.e. moving hand can only move in a certain allowable region (circle) relative to the shoulder. This observation is termed as 'locality of movements'.

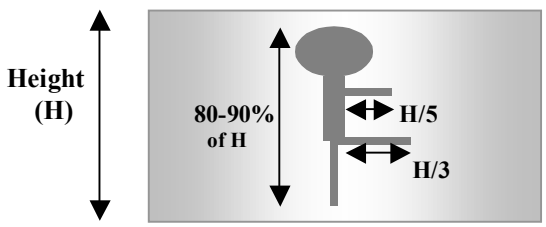

Figure #2: Locality of Movements

If a person does not walk and at any instance his hand is located at point (x , y), then in next frame his hand can be found in the region ($x \pm \Delta x$, $y \pm \Delta y$) with Δx and Δy some defined values.

Experimental Fact: If a height of the person visible in the frame is 80% to 90% of the frame height, person's hand can move at the most H/5 in either direction and his leg can move H/3 in either direction.

3.2. Self-Occlusion

Often for non-rigid bodies, like a human body, one moving part comes in front of other in a frame. This results in the disappearance of other moving part in the next frame. This phenomenon is called Self-Occlusion.

3.3. Range-based Mode Filter (RM Filter)

The range-based mode filter is used to check if a pixel block is a part of moving object or not. Each pixel is compared with object sensor color range. If more than fifty percent pixels in the block are in the object sensor color range, the block is a part of the object. This range-based comparison allows for variations in sensor color during image acquisition.

4. System Architecture Description

Five Processing Agents shown in Figure 1 are explained below.

4.1. System Initialization Agent (SIA)

This agent identifies and stores the initial position, band colors and states of hands and legs in the History. The end-user provides the average color value of each object by some mechanism (by clicking on the initial frame). SIA divides the initial frame into square blocks of size (say 5x5) and using Range Based Filter labels each block as part of some moving object or background. After complete processing of the frame there will be blocks labeled as right-leg, left-leg and so on. A block-merging algorithm is used to merge adjacent blocks having the same label. In the end only one block per hand / leg remains and its center represent the initial position of that part.

Finally the color information, initial positions and initial states (which is by default un-puched in the case of

hands and un-kicked in the case legs) of hands and legs are stored in the history

4.2. Proximity Calculation Agent (PCA)

Suppose a hand in un-punched state. According to LoM, it can move a maximum H/5 in either direction. The allowable region for this hand will be a square having dimensions 2H/5 x 2H/5. Thus if a hand lies at (X, Y) in a frame, it can be found in the region (X ± H/5 , Y ± H/5) in the next frame.

By processing the full allowable region of 2H/5x2H/5, the probability of locating a hand in next frame is maximized. But it results in many pixels being processed multiple times and thus hinders real-time performance. Experiments have shown and the nature of application is such that the search region for a hand can be limited by applying some heuristics and facts (locality of movement) thereby resulting in better frame processing rate. The experimental results have shown that ±H/9 range in x-axis and ±H/18 range in Y-axis give optimal results.

Similarly for tracking the movements of legs experimentally obtained optimal ranges of the search window are ±H/7 in x-axis and ±H/14 in y-axis. Usually PCA calculates the proximity for motion detection without getting any help from other agents. But in certain cases the Self-Occlusion Detection Agent (SODA) instructs it to incorporate certain parameters in the calculation of a proximity region. This happens when Object Detection Agent is unable to detect an object in a proximity region and SODA is initiated to identify the problem and adopt some remedial action for next few frames. This may result in increasing the Search Region for that object.

4.3. Object Detection Agent (ODA)

Object Detection Agent tries to locate the object in the region given by the Proximity Calculation Agent. It applies Range Mode filter to each block (of size 5x5) in the region and labels it as part of object or background. After passing RM filter over the whole region it runs the Block-Merging algorithm. This algorithm merges adjacent blocks that are part of the object into a bigger block and determines the minimum-enclosing rectangle for that object. Then the final size of the unified blocked is checked against two threshold values that represent the maximum and minimum sizes of a hand / leg. If unified block has size between these two thresholds then the object has been detected successfully otherwise block represents background or some noise.

If object detection is successful then the coordinates of unified block are stored in history and the classification agent is initiated to classify the new position into some defined state. If it is unsuccessful in identifying the object, it initiates the self-occlusion detection agent (SODA) for approximating the position of unidentified object.

The size of the range mode filter has been chosen to be 5x5. Choosing a small size 3x3 induces noise. Through experiment we found that filter of size 5x5 and 7x7 best suites our needs.

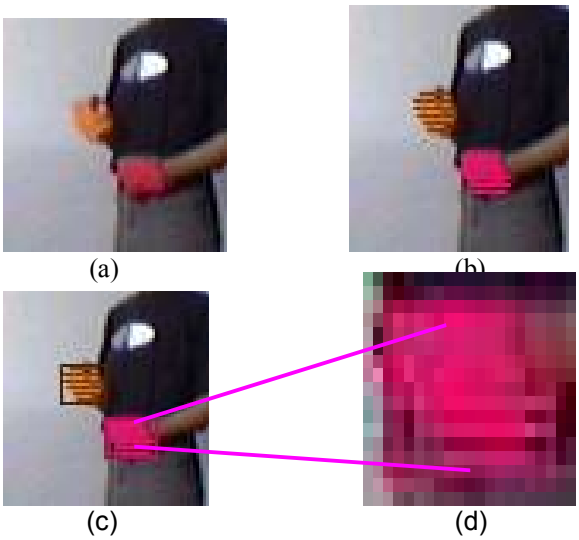

Figure # 3: (a) Captured Image, (b) Color Rows are identified (c) Color rows are merged into one, (d) Bigger view of merged rows

4.4. Self-Occlusion Agent

This agent is initiated whenever Motion Detection agent has failed to identify an object in a given frame. The proximity region for the object and the frame acts as the input to this agent. The main objective of this agent is to decide whether self-occlusion has occurred or not and then to take remedial actions in either case.

Different approaches have been identified for identifying the self-occluding objects [5]. For this application requiring real-time performance, a simplified mechanism is proposed. It uses the previous history of the object and requires that frame-processing rate should be considerably faster (double or more) than the speed of body parts.

Following steps summarizes the overall work:

1. Estimate next position of the object.
2. Decide whether self-occlusion has occurred in certain body parts or not.
3. Calculate remedial actions for next frames.

The next move can be estimated from the previous history of that object. By processing the previous few frames and by calculating the periodicity in the

movement of that object (if some exists), the next location of the object can be estimated. One way can be to process few previous frames (say 5 frames) and try to find the direction vector in subsequent frames in x and y directions. Using some heuristic suitable to the target application, the next position of the object can be predicted with considerable certainty.

After calculating the potential next position (PNP), decision is taken about whether self-occlusion has occurred. Here two factors are considered: one whether potential next position lies in the proximity region used by Motion Detection Algorithm. Second whether some other object has been identified in the near vicinity.

Potential Next Position	*Another object in Vicinity*	*No other object in Vicinity*
In proximity region	Self-Occlusion has occurred	Indecisive. Factors like intensity variation, rotation in object, etc have effected the input image
Not in proximity region	Possibility of Self-Occlusion	Self-Occlusion has not occurred. Object out of range. Check in next frame.

4.5. Classification Agent (CA)

The main purpose of this agent is to identify the current state of a hand / leg. It also identifies whether change in state with respect the previous frame has occurred or not. Using current location and the previous location of a hand/leg, infinite ways of interpretations can be established; each using different domain knowledge. To define states in the form of punched / un-punched and kicked / un-kicked, following interpretation of hands and legs position can be utilized.

Each hand /leg has an allowable region for movement. We can divide this region in different portions with each portion representing a specific state of the object. One interpretation could be to divide the horizontal range into 3 parts. Following figure shows the visual representation of this interpretation.

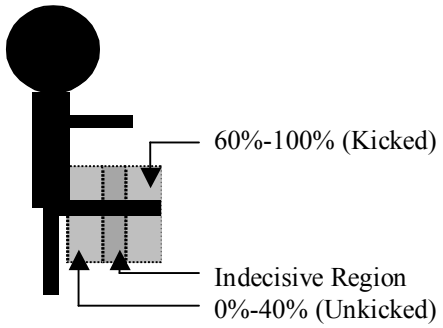

Figure # 4: Classification Ranges

If the current horizontal position is less than the 40% of the allowable range in x-direction, then it represents the un-kicked state. If horizontal position is more than 60% of the maximum horizontal range, then this state can be labeled as kicked. Otherwise leg is between these two states and will be labeled as indecisive until it moves into other two defined states.

5. Results And Conclusion

The mechanism used for motion detection is very simple and contains few very interesting features.

1. ***Light Intensity Variation:*** Because of Range-based Mode (RM) filter, the object detection algorithm can tolerate certain degree of light intensity variation without compromising on the performance. To improve further performance, the Hue, Saturation, Luminosity color-scheme is recommended rather than using RGB.
2. ***Deformation in Object:*** During merging of smaller blocks in to a bigger one, the minimum enclosing rectangle is calculated and the center points of these rectangles are considered for motion detection and classification. Therefore the deformation in the moving object will not have considerable impact on the tracking and classification of objects.
3. ***Rotation in Object:*** Classification algorithm checks only difference of coordinates in x-direction without considering the direction. This gives flexibility to rotate the human body at an angle of 180±10 degree with out effecting the performance.

Using 5*5 RM filter size, and color ranges between 15-30, the computer simulation of above mechanism on a P-III machine has given following results

	Detection Rate
Only hands	*90 % +*
Both hands and legs	*65 % +*

6. References

[1] L. Zhao, C Thorpe, "Stereo and Neural Network-Based Pedestrian Detection". ITSC 99, Tokyo , Oct 5-10, 1999

[2] Y. Song, X. Feng, P. Perona , "Towards Detection of Human Motion"CVPR 2000, June 2000.

[3] K. S. Bhat, M. Saptharishi, P.K. Khosla, "Motion Detection and Segmentation Using Image Mosaics" Carnegie Mellon University

[4] A. J. Lipton, “Local Application of Optic Flow to Analyse Rigid versus Non-Rigid Motion”. Paper Number CMU-RI-TR-99-13, 1999, CMU, USA

[5] J. M. Rehg, T. Kanade, “Model-Based Tracking of Self-Occluding Articulated Objects”. 5th Int. Conf. on Computer Vision, Cambridge, June 1995.

[6] J. M. Rehg, T. Kanade, “DigitEyes: Vision-Based Hand Tracking for Human-Computer Interaction”. IEEE Workshop on Motion of Non-Rigid and Articulated Objects, Austin, Texas, November 1994.