Pivot-Object Motion Detection using Outcode - Nonatree

Pallavi Upadhyay
Lachoo Memorial College of Science & Technology, Jodhpur

D K Sharma
Department of Physics, JNVU, Jodhpur

Mukesh Upadhyay
Lachoo Memorial College of Science & Technology, Jodhpur

ABSTRACT
Motion gaming and hand or body gestured controlled systems (Augmented Reality) is the future ahead, where the movement of hands or movement of pivot-object is used for exerting the controls, eliminating the need of joysticks and control pads.

In this paper we propose Outcode-Nonatree (OCNT), as a new methodology of identifying the motion of pivot-object in vicinity of camera. A live image of pivot-object is captured via camera; it is divided into 9 partitions (Nona-tree Partitions), labeled, using outcode. Then, recursively these partitions are scanned for appearance of pivot-object, till the partitioning fits approximately to the object size. This partition then defines the current position of the pivot-object, used to take the motion decisions.

The proposed methodology is applied for learning, memorizing and then identifying the position of the pivot-object.

Keywords
Motion Gaming, Augmented Reality, Outcode, Nonatree, Neural Networks.

1. INTRODUCTION
The control systems in gaming, simulators, industrial machine control, controlled robotic movement, medical demonstrations to name a few, uses joystick or control pads to generate the control signals for taking the motion related decisions. In this paper we propose a new methodology of exerting the control by using the pivot-object: a specific color-shape object whose movements are used as coordinated output for the purpose of control. The control system comprises of a camera as capture device, observing the movement of the pivot-object continuously, processing the pivot-position, generating the control output. The method has three phases to carry out; firstly, to learn the pivot-object, secondly, to memorize the current position of the object and finally tracking the motion of the pivot-object and based on defined control scheme in the system, to generate or exert the control as output. The pivot-object should be a specific color object and should have different color as of the background. Typically in our experiment we have considered Red color ball as pivot-object.

2. NONATREE-OUTCODE
The visual area of control, under the vicinity of the camera is partitioned into nine segments using outcode [Cohen-Sutherland] techniques as: Above, Below, Right and Left codes typically called ABRL code. These nine segments are the area of search of pivot-object forms a Nonatree as illustrated in Figure- 1.

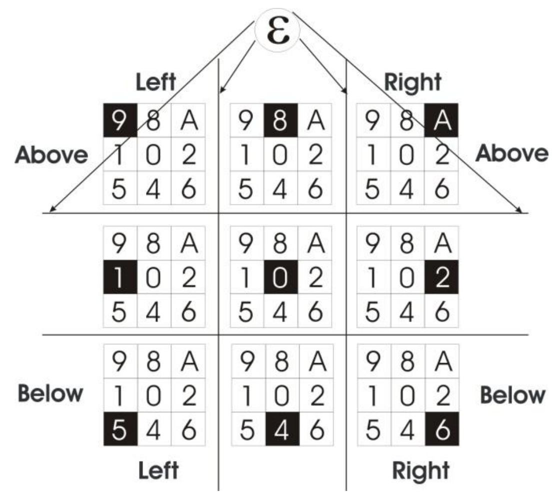

Figure 1: Outcode-Nonatree

The outcode follows binary weights for each of the partition line, with A=8, B=4, R=2 and L=1. Thus each of the partition uses 4 bit binary code with weights as ABRL.

The capture window is defined by W_X= (W_{Xmin}, W_{Ymin}) and W_Y= (W_{Xmax},W_{Ymax}). The typical value in our experiment is (0, 0)-(320,240), one of the standard window of camera image. The partition lines for segmentation of capture window are computed using Table-1,

Table-1: Object Partition Coding

Code AB	Y-Coord.	Code RL	X-Coord.
00	**1/3**	**00**	**1/3**
01	**2/3**	**01**	**0**
10	**0**	**10**	**2/3**
11	**NA**	**11**	**NA**

For example, the partition with code 4 has a binary value of 0100, and with reference to above table the value of y-coordinate is 2/3 of W_{Ymax}-W_{Ymin} . and similarly the value of x-coordinate is 1/3 of W_{Xmax}-$W_{Xmin.}$ Figure 2 shows the coordinate of segments of the window.

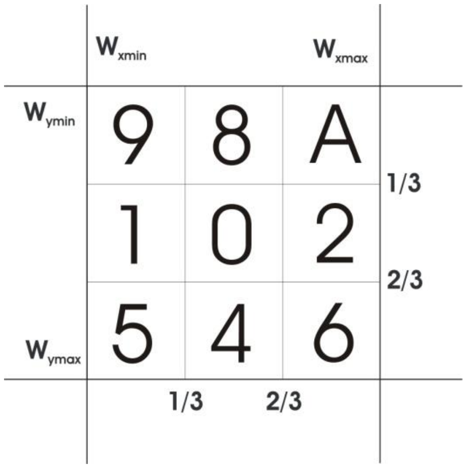

Figure 2: Partitioned Coordinate Window

3. PIVOT-OBJECT LEARNING, MEMORIZING & TRACKING

The pivot-object is the basis to motion control. Initially, in learning phase the pivot-object is kept at a stable position with a condition that the background (ambience) is not having any color as of the pivot-object. The distance of pivot-object should be such that its image formation inside the capture window should be less than the segment size of some k^{th} level. The level k is the depth of partitioning. The level k is considered maximum to a limit where the depth of Nonatree partitioning reaches to a condition wherein the size of partition almost fits the size of pivot-object.

At some stable position of pivot-object, the image is captured and it is processed by counting the numbers of pivot-object color pixels.

The captured image is now processed with Nonatree partitioning, producing nine segments. Each segment turn-by-turn is processed with *pixel-count algorithm*, as shown in Figure 3, generating the percentage of presence of pivot-object. With reference to the pixel percentage presence each segment is labeled as E, P, or F (Empty, Partial or Full).

a. **E**: if there is no presence of pivot-object pixel, this partition is not considered for further processing.

b. **P**: if the pivot-object pixels are present, in the partition. Here, there can be two cases

i) The object is partially inside the partition

ii) The object is completely inside the partition, but the pivot-object pixel count is less than further partition size of the segment background color pixel count.

Under both the above conditions this partition is considered for further processing.

c. **F**: if the pivot-object pixels tightly equal the segment pixel counts. This is also the terminating condition of the processing.

3.1 Pixel Count Algorithm

The algorithm accepts the partition window coordinates as input and returns back the specific color pixel count.

```
Function CountPix(x1,y1,x2,y2:Integer): Word;
Var x,y : Integer;
colorvalue : Longint;
Begin
  CountPix := 0;
  For y := y1 to y2 do
   For x := x1 to x2 do
   Begin
     colorvalue:=
ImageMain.Picture.Bitmap.Canvas.Pixels[x,y];
     Rval := GetRVAlue(Colorvalue);
     Gval := GetGVAlue(Colorvalue);
     Bval := GetBVAlue(Colorvalue);
  If (Rval > 0) and (Gval =0) and (BVal =0) then
       ColorCount := ColorCount + 1;
  End;
End;
```

Pixel Count Algorithm

3.2 Coordinate Computation of Pivot-Object

Current co-ordinate position of the pivot-object is computed with reference to the level of recursion, which terminates further partitioning. Each Nona-tree formed segment is available with E, P or F label. The P label partition is further processed till one of the partition is F or a terminating condition is reached.
The number sequence generated provides the co-ordinate position of the pivot-object.

For example: 941: it is a three level code, meaning that the algorithm first digit 9 represents first level partition code (above left), the second digit 3 represents second level partition code (center below) and, the third digit 1 represents third level (center left)
Thus the pivot-object position is

1. First level
   a. $aX1 = 0$, $aX2 = 1/3 * W_X$
   b. $aY1 = 0$, $aY2= 1/3 *W_Y$
2. Second level
   a. $bX1 = 1/3* aWx$, $bX2 = 2/3 * aWx$
   b. $bY1 = 2/3*aWy$, $bY2 = aY2$
3. Third level
   a. $cX1 = bX1$, $cX2 = 1/3*bWx$
   b. $cY1 = 1/3*bWy$, $cY2 = 2/3*bWy$

Note: The prefix before coordinates defines level, namely, a: First level, b: Second level, c: Third level and so on.

The pivot object under vicinity of camera is learned by the network and then follows the motion of the object. The whole methodology is then automated by developing suitable software. This automation increases the efficiency of identification of the random object motion providing the coordinates of presence in the window of capture.

4. SIMULATION & MOTION DETECTION

To verify the methodology as defined we have developed software with a bitmap image formed with resolution of 320x240, having a Red colored pivot-object placed at the center of the image, as shown in Figure 3.

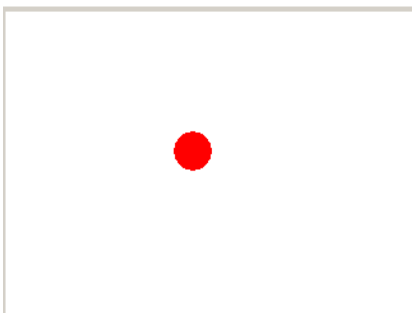

Figure 3: Pivot-object Image

This image is then processed with Nonatree partitioning, applying the pixel count algorithm. In our experiment for the first level partitioning segment-0, is F and remaining segments are E. Thus, segment-0 is considered for further partitioning. During the second level Nonatree partitioning of segment-0, the computations of pivot-object pixel presence are as: Partition 0: 46%, Partition 8: 28%, Partition 1: 16% and Partition 9: 10%. At this level, since the pivot-object pixel count is nearly equal to the partition pixel count hence the process is terminated, thus partition-0 of second level is the position of the pivot-object. Figure 4 illustrates two level partitioning.

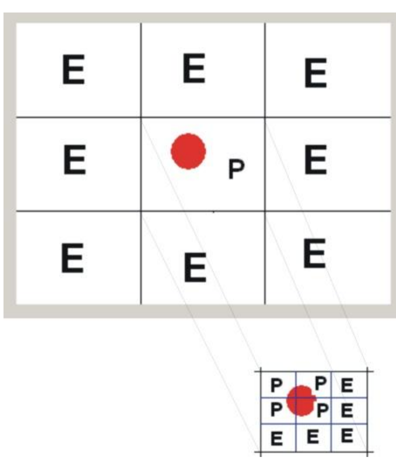

Figure 4: Two level Partitioning

Thus, the final position of pivot-object is available after 2nd level of processing as shown in Figure 5.

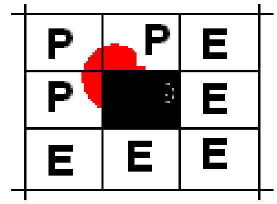

Figure 5: Final Position of Pivot-object

5. SCOPE FOR FUTURE WORK

This methodology can be carried out by training the Neural-Networks, whereby, the network will automatically learn the feature of pivot-object adjusting the neurons weight according to the level of partition and pivot-pixel count. Further, for the purpose of tracking the neurons will be dynamically trained depending on the level of recursion required to identify the position of the pivot-object. Also, in further extension 3 Dimensional movement of the pivot-object can be computed using three image capturing devices.

6. BIBLIOGRAPHY

[1] T. Lakhavijitlert T and K. Prachumrak., "*Nona-Tree Weighted Finite Automata Compression of Simple Images*", World Academy of Science, Engineering and Technology 21 2006

[2] Yih-Kai Lin and Hsu-Chun Yen, ."*An ω-Automata Approach to the Representation of Bilevel Images., IEEE Transaction on Systems, Man, and Cybermetics*"-Part B, vol 33, 2003, June 2003, pp. 524-531.

[3] F.Katritzke, W. Merzenich, M. Thomas, "*Enhancements of partitioning techniques for image compression using weighted fintie automata*", *Elsevier*, 2003.

[4] D. Xiao and R. Hubbold. "*Navigation guided by artificial force fields*" . In ACM CHI 98 Conference on Human Factors in Computing Systems, Los Angeles, CA, USA, April 1998.

[5] Gregory M. Hunter, and Kenneth Steiglitz, " *Operations on Images using Quad Trees*", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAM-1, No.2, April 1979.

[6] James D. Foley et.al. "*Computer Graphics, Principles and Practice*", Pearson Education, 1996.

[7] Donald Hearn and M. Pauline Baker, " *Computer Graphics*", 2nd Edition, Eastern Economy Edition, 1998.