Int. J. Communications, Network and System Sciences, **2009, 7, 687-694**
doi:10.4236/ijcns.2009.27079 Published Online October 2009 (http://www.SciRP.org/journal/ijcns/).

Augmented Reality for Realistic Simulation Using Improved Snake and Picking Algorithm by Proportional Relational Expression

JeongHee CHA[1], GyeYoung KIM[2], HyungIl CHOI[3]
[1]*Division of Computer Information, School of Computing, School of Media,*
[2]*BaekSeok Culture University,Anseo Dong, DongNam Gu, Cheonan, Korea,*
[3]*Soongsil University , Sangdo 5 Dong , DongJak Gu, Seoul, Korea,*
Email: pelly@bscu.ac.kr, *{gykim11,hic}@ssu.ac.kr*
Received May 8, 2009; *revised June* 29, 2009; *accepted August* 17, 2009

ABSTRACT

In realistic simulation of mobile Augmented Reality, essential point is how to best depict occluded area in such a way that the user can correctly infer the depth relationships between real and virtual objects. However, if the constructed 3D map of real world is not accurate or the density is not sufficient to estimate the object boundary, it is very difficult to determine the occluded area. In order to solve this problem, this paper proposes a new method for calculating the occlusion area using the improved snake algorithm and picking algorithm by the proportional relational expression. First, we generated the wireframe by the DEM in the experimental region and mapped to CCD real image using visual clues. And then, we calculated the 3D information at the point where occlusion problem for a moving virtual target by the proposed method. Experimental results show the validity of the proposed approach under the environment in which partial occlusions occur.

Keywords: Snake, Picking, DEM, Occlusion, AR, Simulation

1. Introduction

This paper studied on the development of a realistic simulated training model through the display of virtual targets in the input images of CCD camera mounted in a vehicle and the determination of occlusion areas generated from the creation and movement of virtual objects through a movement path according to a scenario. Augmented reality has three general characteristics: image registration, interaction, and real time [1,2]. Image registration refers to the matching of the locations of the real world object that user watch and the related virtual object. Interaction implies that the combination of virtual objects and the objects in real images must be harmonized with surrounding environment in a realistic manner, and refers to the determination of occlusion areas according to the changed location or line of sight of the observer or the re-rendering of virtual objects after detection of collisions. Real time refers to the real time image registration and interaction. A key problem in the AR field is how to best depict occluded objects in such a way that the viewer can correctly infer the depth relationships between different real and virtual objects. For occlusion processing such as the hiding of farther virtual objects by closer real objects and the covering of objects in real images by other virtual objects, the two worlds must be accurately coordinated and then the depth of the actual scene must be compared with the depth of virtual objects [3,4]. But if the accuracy or density of the created map is insufficient to estimate the boundary of occlusion area, it is difficult to determine the occlusion area. To solve this problem, first, we created a 3D wireframe using the DEM of the experiment area and then coordinate this to CCD camera images using visual clues. Second, to solve the problem of occlusion by accurately estimating the boundary regardless of the density of map, this paper also proposed a method to obtain the reference 3D information of the occlusion points using the improved Snake algorithm and the Picking algorithm and then to infer the 3D information of other boundaries using the proportional relations between 2D and 3D DEM. Third, for improving processing speed, we suggest a method by comparing the MER (Minimum Enclosing Rectangle) area of the real object in the camera's angle of view and

the MER of the virtual target.

We briefly review related work in Section 2. In Section 3, we outline the framework of our proposed algorithm. The methodology of Wireframe modeling, improved snake algorithm for extracting image boundary and picking algorithm for acquiring 3D information are explained in Section 4. Section 5, we show the experimental results and the validity of the proposed approach. Finally, in Section 6 we discuss drawbacks of the algorithm and propose possible future work.

2. Previous Work

A basic design decision in building an AR system is how to accomplish the combining of real and virtual. Toward this purpose, many researchers make efforts to minimize virtual objects registration error and to increase the realness of virtual objects [5]. Drastic and Milgram list a number of cues that a user may use to interpret depth, including image resolution and clarity, contrast and luminance, occlusion, depth of field and accommodation [6]. We can use one of two technologies to see the real world in AR, one is optical see-through and the other is video see-through. These technologies can present occluded objects, and each has a variety of challenges [7]. Blurring also can help compensate for depth perception errors [8]. Koch uses computer vision techniques to infer dense and accurate depth maps from image pairs, and uses this information to construct 3D graphical representations of the restricted static environment [9]. Several authors observe that providing correct occlusion of real objects by virtual objects requires a scene model. Correct occlusion relationships do not necessarily need to be displayed at all pixels. The purpose of many applications is to see through real object. We introduce here mobile vehicle-mounted display system capable of resolving occlusion between real and virtual objects. We restricted the real environment to some area and constructed that area's scene Model using 3D information. The heart of our system is boundary extraction algorithm and 3D information inference algorithm of object boundary. Figure 1 shows our monitor-based training vehicle configuration.

In our experimental vehicle configurations, we send steering, acceleration, brake data to car driving controller through Bluetooth using remote car controller. Vehicle can be controlled by transmitted data and we can get feedback of present car location data by mounted sensor system. RS232 communicator is interface between vehicle driving controller and sensor fusion system. And it receives instructions from sensor system. CCD camera views the environment. The camera may be static or mobile. In mobile case, the camera might move around by being attached to a vehicle, with their locations tracked by GPS and INS. The image of real world and the virtual images generated by a scene generator are combined.

3. System Flow Description

Figure 2 outlines the framework of our proposed system. System has two stages. First stage is environment map construction stage. It consists of registration of two world using visual clues and object contour extraction and acquiring 3D information. Second stage is virtual object rendering stage. It has creation of virtual target path and selection of candidate occlusion object and occlusion processing.

4. Methodology

4.1. Formation of Wireframe Using DEM and Registration with Real Images Using Visual Clues

The topographical information DEM (Digital Elevation

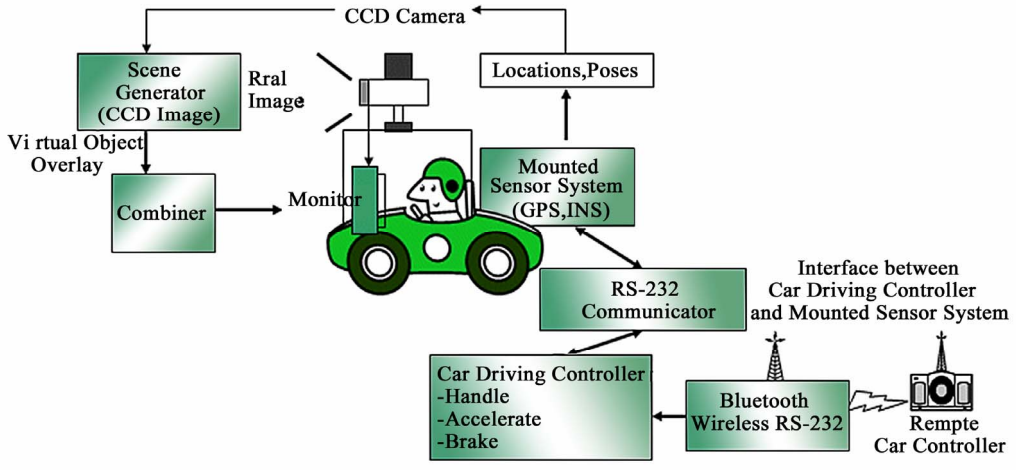

Figure 1. Training vehicle configuration.

Model) is used to map the real world coordinates to each point of the 2D CCD image. DEM has information on the latitude and longitude coordinates expressed in X and Y and heights in fixed interval. The DEM used for this experiment is a grid-type DEM which had been produced to have the height information for 2D coordinates in 1M interval for the limited experiment area of 300 m x 300 m. The DEM data are read to create a mesh with the vertexes of each rectangle and a wireframe with 3D depth information [10,11] as Figure 3. This is overlaid on the sensor image to check the coordination, and visual clues are used to move the image to up, down, left or right as shown in Figure 4, thus reducing error. Based on this initial coordinated location, the location changes by movement of vehicles were frequently updated using GPS (Global Positioning System) and INS (Inertial Navigation System).

4.2. Extracting the Contour of Objects in Image by Enhanced Snake Algorithm

4.2.1. Edge Map Using Gradient Vector Flow

The Snake algorithm [12,13] is a method of finding the outline of an object by repeatedly moving to the direction of minimizing energy function from the snake vertex input by user. But existing snake algorithm cannot accurately extract the contour information when the object form is complex because as shown in Figure 5, the direction of the energy function appears as a composite vector of the current, previous, and the next snake points, and shrinks toward the center of these points. To solve this problem, this paper proposes a method to form an edge

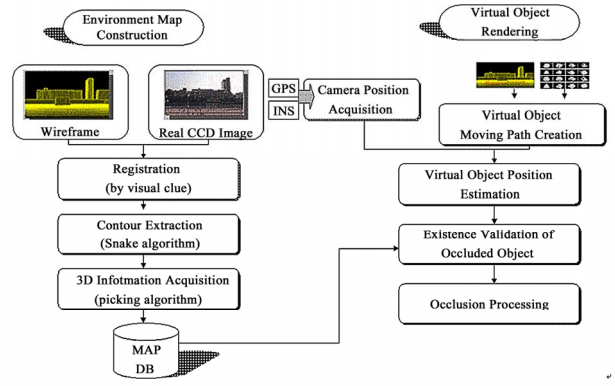

Figure 2. Proposed system framework.

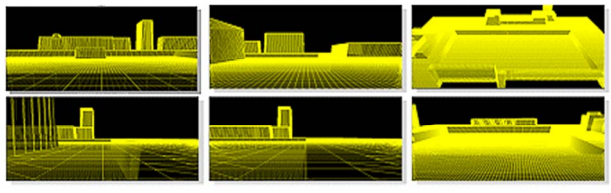

Figure 3. Wireframe creation using DEM.

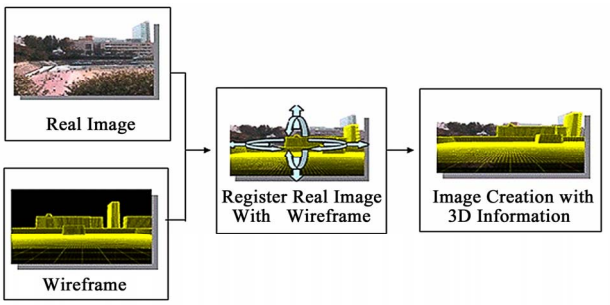

Figure 4. Registration of two worlds using visual clues.

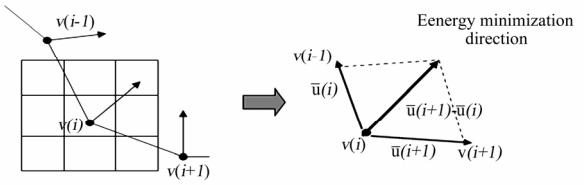

Figure 5. The direction of energy minimization search in snake algorithm.

map using the Gradient Vector Flow (GVF) algorithm [14,15,16], and add a new energy term that indicates the distance between the searched edge point and snake point so as to extract an accurate contour.

The GVF algorithm can measure the contour of complex objects using the gradient of edge, and move to the concave contour regardless of initialization. Further, the gradient vector of the edge map has a larger value as it is near edge, and approaches zero as it is farther. This paper uses the edge information of the gradient vector flow to search the proximal edge point, and when there is an edge, adds a new energy term($E_{edge\text{-}distance}$) that shows the dis tance from the reference point to the searched edge as Equation (1). Here, α β, and γ are all set to 1 without exhaustive adjustment .

$$E_{snake} = \int_0^1 E_{continuity}(v(s)) + E_{curvature}(v(s)) + E_{image}(v(s)) + E_{edge-distance}(v(s))ds \quad (1)$$

4.2.2. The Proximal Edge Search Method

Figure 6 shows a proposed proximal edge search method in this paper.

First it searches edge points while rotating around the axis d which is the connection between current and previous snake points v_i and v_{i-1}. In other words, if the angle formed by the three points v_i, v_{i-1}, and v_{i-2} is ϕ, to prevent the situation where the axis meets with or passes by v_{i-2} and meets v_i again, it searches the edge point v_i' where the image strength ∇I is greater than the threshold while rotating only by $\frac{\phi}{2}$ and adds a new energy term using the value of the distance d' between v_i and v_i' to the existing algorithm. This paper determined the rotation

direction for accurate search by assuming the following two facts: First, it was assumed that the initial snake points form a closed curve that encloses the object. Second, it was assumed that the points were arranged sequentially in one direction. The reason for this was because to search proximal edge, it must move inside the contour, but the direction may be wrong due to the diversity of object forms if simply the direction to the object center is set. Figure 7 is an example of setting the rotation direction of the snake points.

4.2.3. Calculation of $E_{edge\text{-}distance}$

Figure 8 is an example of calculating the distance between an arbitrary snake point v_i and the edge $v_i^{'}$ around it. If we surround the arbitrary point v_i with a 9×9 window and assume that its distance with a new edge is $d_{mn}^{'}$, the height and width of the window are s, and the horizontal and vertical positions of the snake point in the window are m and n, the $d_{mn}^{'}$ can be obtained with the Equation (2) by the Euclidean theorem, and the energy term to be added can be defined as the Equation (3) by applying the distance value instead of the brightness value of the image term.

$$d_{mn}^{'} = \sqrt{\left(\frac{2(|v_x - v_x^{'}| + m) - s + 1}{2}\right)^2 + \left(\frac{2(|v_y - v_y^{'}| + n) - s + 1}{2}\right)^2} \quad \textbf{(2)}$$

$$\begin{aligned} E_{edge-distance} &= (|v_i - v_i^{'}| - d_{\min}^{'}) / (d_{\max}^{'} - d_{\min}^{'}) \\ &= (d_{mn}^{'} - d_{min}^{'}) / (d_{max}^{'} - d_{min}^{'}) \end{aligned} \quad \textbf{(3)}$$

Added new energy term $E_{edge\text{-}distance}$ is expressed together with continuity and curvature energy terms in Figure 9. When only the two terms of the existing algorithm were considered, the minimum point of energy was in line 3, column 5, but the location changed to line 4, column 6 when the energy value in consideration of the distance between proximal edges was included. In conclusion, the flow of the enhanced snake energy function to which the proximal edge energy function is added can extract the edge exactly in complex situations by approaching the edge more closely.

Table 1 shows the pseudo codes of the proposed algorithm using proximal edge search method.

4.3. Acquisition of 3D Information Using the Picking Algorithm

In order to acquire the 3D information of the extracted vertexes, this paper used the Picking algorithm which is a well-known 3D graphics technique [17]. It finds the collision point with the 3D wireframe created by DEM that corresponds to the points in 2D image and provides the 3D information of the points. The picking search point is the lowest point of the vertexes of the objects

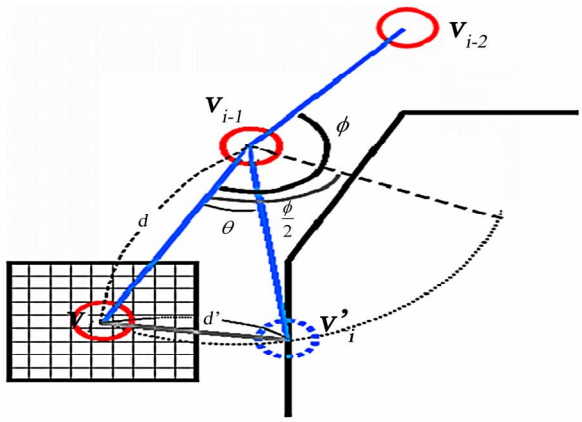

Figure 6. The proximal edge search method.

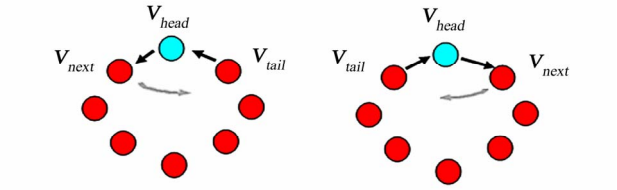

Figure 7. Snake's rotation direction.

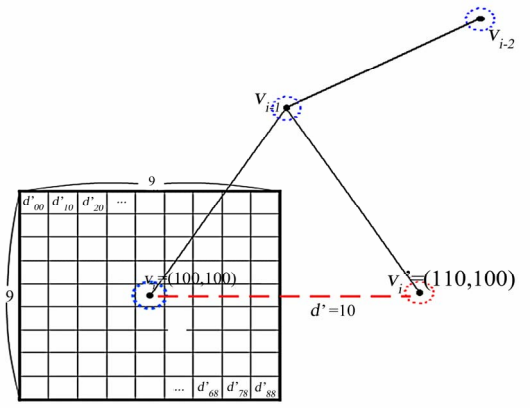

Figure 8. Distance between a point of snake and edge.

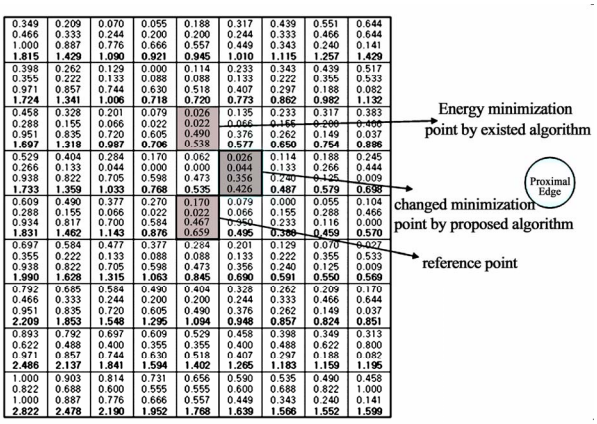

Figure 9. Changed energy minimization point by proposed algorithm.

Table 1. Pseudo codes of proposed algorithm.

```
Do    /* loop for proposed algorithm */
 For i=0 to n-1 /* n is number of snake points */
      Angle = ( ∠ v_{i-2} v_{i-1} v_i ) / 2 ; /* search limit determination */
      for j = 0 to Angle
      if  v_i'  is Edge then bFind = true;
      E_min = BIG ;
      for j = 0 to m-1   /* m is size of neighborhood */
          if bFind is True then
          E_j = E_cont,j + E_curv,j + E_image,j + E_edge-distance,j ;
          Else  E_j = E_cont,j + E_curv,j + E_image,j ;
          If  E_j < E_min  then
              E_min = E_j ;
              j_min = j ;
      move point  v_i  to location  j_min ;
      if ( j_min  != current location) cnt_movedpoint += 1;
      /* following process determines where to allow corners   */
      For i=0 to n-1
          c_i = | u_i / |u_i| - u_{i+1} / |u_{i+1}| |^2 ;
      For i=0 to n-1
      If  c_i < c_{i-1}  and  c_i > c_{i+1}  ;
      /* if ci(curvature) is larger than neighborhood's */
        and  c_i >  threshold1 ;/* if ci is larger than threshold1 */
        and  mag (v_i) > threshold2;
         /* if edge strength is larger than threshold2 */
        Then  β_i =0; /* relax curvature at point i */
        Until cnt_movedpoint < threshold3;
```

extracted from the 2D image. The screen coordinate system that is a rectangular area indicating a figure that has been projection transformed in the 3D image rendering process must be converted to the viewport coordinate system in which the actual 3D topography exists to pick the coordinate system where the mouse is actually present. First, the conversion matrix to convert viewport to screen is used to obtain the conversion formula from 2D screen to 3D projection window, and then the ray of light is lengthened gradually from the projection window to the ground surface to obtain the collision point between

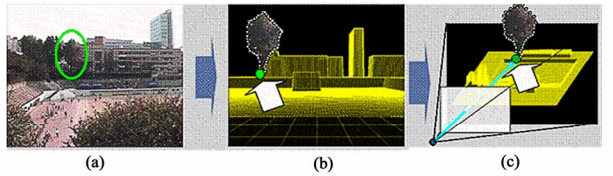

Figure 10. 3D information extraction using collision point of ray and DEM. (a)occlusion candidate (b)matching ref.point and DEM (c)3D information extraction.

the point to search and the ground surface. Figure 10 is an example of picking the collision point between the ray of light and DEM. The lowest point of the occlusion area indicated by an arrow is the reference point to search, and this becomes the actual position value of 2D image in a 3D space.

4.3.1. Creation of 3D Information Using Proportional Relational Expression

The collision point, or reference point, has 3D coordinates in DEM, but other vertexes of the snake indicated as object outline cannot obtain 3D coordinates because they don't have a collision point. Therefore, this paper suggested obtaining a proportional relation between 2D image and 3D DEM using the collision reference point and then obtaining the 3D coordinates of another vertex. Figure 11 shows the proportional relation between 2D and 3D vertexes. In Figure 11, S_m is center of screen, S_B is reference point of snake vertexes (lowest point), $\Delta S_B = (\Delta S_{x_B}, \Delta S_{y_B})$ is a distance from S_m to S_B, S_k is a optional point except reference point of snake vertexes, $\Delta S_k = (\Delta S_{x_k}, \Delta S_{y_k})$ is a distance from S_m to S_k. P_m is a projection point of straight line of P_B in 3D, which is through the center of screen. P_B is a 3D correspondence point of S_B, $\Delta P_B = (\Delta P_{x_B}, \Delta P_{y_B}, \Delta P_{z_B})$, P_k is a optional point except reference point, $\Delta P_k = (\Delta P_{x_k}, \Delta P_{y_k}, \Delta P_{z_k})$, $t = \overline{P_O P_B}$, $t_m = \overline{P_o P_m}$, $\theta_B : \angle t t'$, $\phi_B : \angle t' t_m$. t': a projected vector of t to xz plane.

To get P_m in 3D that passes the center of the screen using the coordinates of the reference point obtained above, t' must be obtained first. As the t value is given by the picking ray, the given t value and y_B are used to get θ_B and t' is obtained using this θ_B in Expression (4).

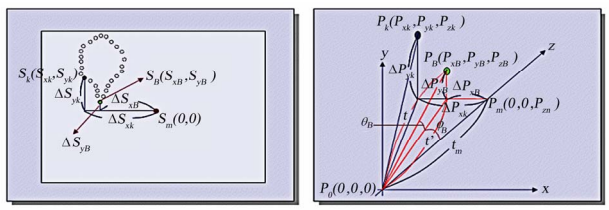

Figure 11. Proportional relation of the vertexes in 2D and 3D.

$$\theta_B = sin^{-1}(\frac{\Delta P_{y_B}}{t}), t' = |t_B| cos(\theta_B), (t' = |t'|) \quad \textbf{(4)}$$

To get t_m, Φ_B which is the angle between $t'$and t_m is obtained, t_m can be obtained using Φ_B from Expression (5)

$$\varphi_B = tan^{-1}(\frac{\Delta P_{x_B}}{t'}), t' = |t_m| cos(\varphi_B) |t_m| = \frac{|t'|}{cos(\varphi_B)} t_m = |t_m| \quad \textbf{(5)}$$

Because t_m=PZ_m, we can define P_m=(0,0, t_m).

Now, we can present the relation between the 2D screen view in Figure 11 and the 3D space coordinates, and this can be used to get P_k, which corresponds to the 2D optional snake vertex.

$$\begin{aligned} &\Delta S_B : \Delta P_B = \Delta S_k : \Delta P_k, \\ &\Delta S_{x_B} : \Delta P_{x_B} = \Delta S_{xk} : \Delta P_{x_k} \\ &\Delta P_{x_k} = \frac{\Delta P_{x_B} \times \Delta S_{x_k}}{\Delta S_{x_B}}, \Delta S_{y_B} : \Delta P_{y_B}, \\ &\Delta S_{yk} : \Delta P_{y_k}, \Delta P_{y_k} = \frac{\Delta P_{y_B} \times \Delta S_{y_k}}{\Delta S_{y_B}} \end{aligned} \quad \textbf{(6)}$$

Consequently, we can get $\Delta P_k = (\Delta P_{x_k}, \Delta P_{y_k})$, which is the 3D point corresponding to each snake vertex to search.

4.3.2. Creation of Virtual Target Path and Selection of Candidate Occlusion Objects Using MER (Minimum Enclosing Rectangle)

To test the proposed occlusion-resolving algorithm, we created the movement path of a virtual target, and determined the changes of the direction and shape of the target as well as the 3D position of the target. First, the beginning and end points of the target set by instructor were saved and the angle of these two points was calculated, and the direction and shape of the target were updated in accordance with the change of the angle. Further, the remaining distance was calculated using the speed and time of the target, and the 3D coordinates corresponding to the position after movements were determined. We also suggest a method of improving processing speed by comparing the MER (Minimum Enclosing Rectangle) area of the object in the camera's angle of vision and the MER of the virtual target because the relational operations between all objects extracted from the image for occlusion processing and the virtual target take much time. The MER (Minimum Enclosing Rectangle) of an object refers to the minimum rectangle that can enclose the object and determines the object that has an overlapping area by comparing the objects in the camera image and the MER of the virtual target. In addition, the distance between object and virtual target is obtained using the fact that the determined object and virtual target are placed more or less in a straight line from the camera, and this value was used to determine whether there exists an object between the virtual target and the camera.

5. Experimental Results

Figure 12(up) shows movement path of the virtual target which trainee sets. Also, (down) shows the various virtual targets created to display the targets changing with movement on the image.

Figure 13, Figure 14 compares the search results, accuracy

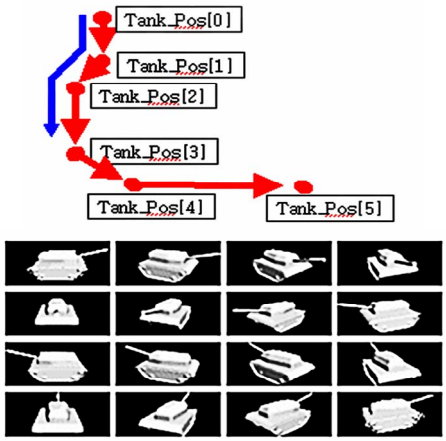

Figure 12. Moving route creation(up) and appearance of virtual object as it moved(down).

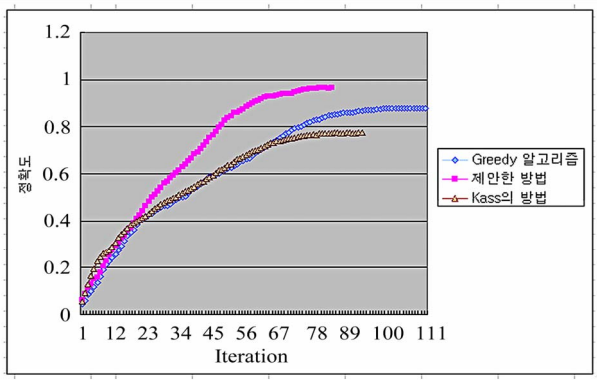

Figure 13. Accuracy comparison(leaf).

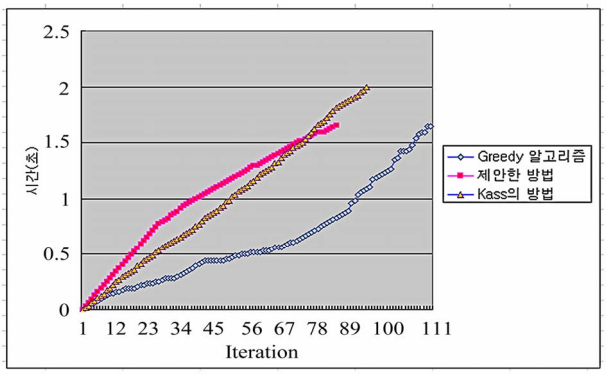

Figure 14. Speedy comparison(leaf).

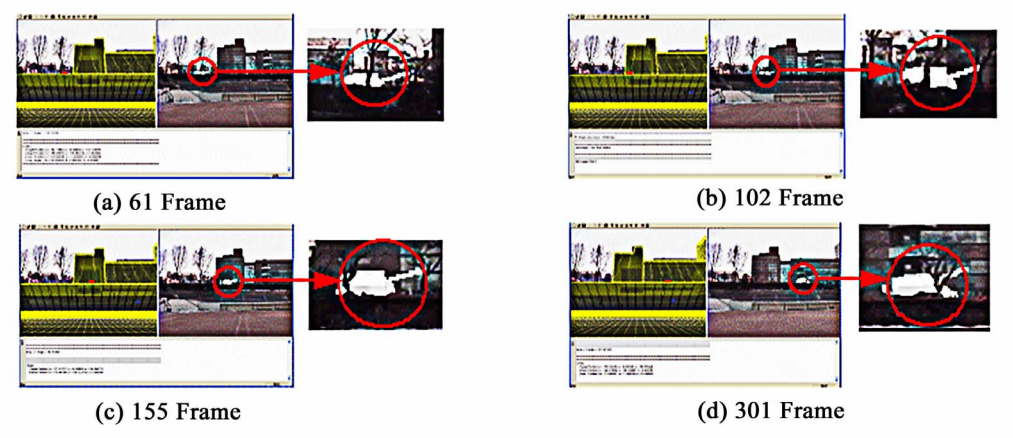

Figure 15. Experimental results of moving and occlusion.

and speed for more complex leaf. As shown in the figures and graphs, we can see that the proposed algo rithm has much higher accuracy and less repetition counts, and the speed is equal to greedy algorithm.

As shown in Figure 13, the proposed algorithm stopped search at the 80th round, and the accuracy was 0.96 while the Kass and greedy algorithms showed the search count 96 and 150 and the accuracy 0.78 and 0.84, respectively. Therefore, we can conclude that the proposed algorithm has higher performance than existing algorithms. The search speed of the proposed algorithm was 1.65 seconds, which is equal level to the greedy algorithms.

Figure 15 shows the virtual images moving along the path by frame. We can see that as the frames increase, it is occluded between the tank and the object.

Table 3 compares between the case of using snake vertexes to select objects in the image to compare with virtual targets and the case of using the proposed MER. With the proposed method, the processing speed decreased by 1.671, which contributed to performance improvement.

6. Conclusions

To efficiently solve the problem of occlusion that occurs when virtual targets are moved along the specified path over an actual image, we created 3D virtual world using DEM and coordinated this using camera images and visual clues. Moreover, the enhanced Snake algorithm and the Picking algorithm were used to extract an object that is close to the original shape to determine the 3D information of the point to be occluded. To increase the occlusion processing speed, this paper also used the method of using the 3D information of the MER area of the object, and proved the validity of the proposed method through experiment. In the future, more research is required on a more accurate extracting method for occlusion area that is robust against illumination as well as on the improvement of operation speed. We also hope to study more in real time environment and to overcome complicated factors that were beyond our control, such as sensor error in the current settings, the brightness difference of same image.

Table 3. Speed comparison.

Method	Total frame	Used object	Speed(sec)	Frame per sec.
Snake vertexes	301	10	112	2.687
MER(proposed)	301	10	67	4.492

7. Acknowledgement

This Work was supported by the Korea Research Foundation Grant (KRF-2006-005-J03801) Funded by Korean Government.

8. References

[1] R. Azuma, "A survey of augmented reality," in ACM SIGGRAPH'95 Course Note #9-Deveoping Advanced Virtual Reality Applications, August 1995.

[2] O. Bimber and R. Raskar, "Spatial augmented reality: A modern approach to augmented reality," Siggraph, Los Angeles, USA, 2005,

[3] J. Y. Noh and U. Neumann. "Expression cloning," In SIGGRAPH'01, pp. 277–288, 2001.

[4] E. Chen. "QuickTime VR—An image-based approach to virtual environment navigation," Proc. of SIGGRAPH, 1995.

[5] A. Ronald and G. Bishop. "Improving static and dynamic registration in an optical see-through HMD," Proceedings of SIGGRAPH'94, Orlando, Florida, In Computer Graphics Proceedings, Annual Conference Series pp. 197–204, July 24-29, 1994.

[6] D. Drastic and P. Milgram, "Perceptional issues in aug-

mented reality," In M. T. Bolas, S. S. Fisher, and J. O. Merritt, editors, SPIE Volume 2653: Stereoscopic Displays and Virtual Reality Systems ///, pp. 123–134, January-February 1996.

[7] J. P. Rolland and H. Fuchs, "Optical versus video see-through head-mounted displays in medical visualization." Presence: Teleoperators and Virtual Environments, Vol. 9, No. 3, pp. 287–309, June 2000.

[8] A. Fuhrmann, G. Hesina, F. Faure, and M. Gervautz, "Occlusion in collaborative augmented environments," Computers and Graphics, Vol. 23, No. 6, pp. 809–819, 1999

[9] K. Reinhard, "Automatic reconstruction of buildings from stereoscopic image sequences," In R. J. Hubbold and R. Juan, editors, Eurographics'93, Eurographics, Blackwell Publishers, Oxford, UK, pp. 339–350, 1993..

[10] S. Growe, P. Schulze, and R. Tnjes, "3D visualization and evaluation of remote sensing data," Computer Graphics International'98 Hanover, Germany, June 22–26, 1998.

[11] E. Chen. "QuickTime VR—An image-based approach to virtual environment navigation," Proc. of SIGGRAPH, 1995.

[12] L. L. Ji and H. Yan, "Attractable snakes based on the greedy algorithm for contour extraction," Pattern Recognition 35, pp. 791–806, 2002.

[13] C. C. H. Lean, A. K. B. See, and S. A. Shanmugam, "An enhanced method for the snake algorithm," First International Conference on Innovative Computing, Information and Control (ICICIC'06), Vol. 1, , pp. 240–243, 2006

[14] C. Xu and J. L. Prince, "Gradient vector flow: A new external force for snakes," Proc. IEEE Conf. on Comp. Vis. Patt. Recog. (CVPR), Los Alamitos: Comp. Soc. Press, pp. 66–71, 1997.

[15] C. Y. Xu and J. L. Prince, "Snakes, Shapes, and Gradient vector fow," IEEE Transactions in Image Processing, Vol. 7, No. 3, Mar. 1998.

[16] C. Y. Xu and J. L. Prince, "Generalized gradient vector flow external frces for active contours," Signal Processing, Vol. 71, No. 2, pp. 131–139, Dec. 1998.

[17] S.-T. Wu, M. Abrantes, D. Tost, and H. C. Batagelo, "Picking and snapping for 3D input devices," In Proceedings of SIBGRAPI'03, pp. 140–147, 2003.

