2007 IEEE International Conference on
Robotics and Automation
Roma, Italy, 10-14 April 2007

Humanoid Robot Painter: Visual Perception and High-Level Planning

Miti Ruchanurucks, Shunsuke Kudoh, Koichi Ogawara, Takaaki Shiratori, and Katsushi Ikeuchi
Institute of Industrial Science, The University of Tokyo, Japan
{miti, kudoh,ogawara,siratori,ki}@cvl.iis.u-tokyo.ac.jp

Abstract **– This paper presents visual perception discovered in high-level manipulator planning for a robot to reproduce the procedure involved in human painting. First, we apply a technique of 2D object segmentation that considers region similarity as an objective function and edge as a constraint with artificial intelligent used as a criterion function. The system can segment images more effectively than most of existing methods, even if the foreground is very similar to the background. Second, we propose a novel color perception model that shows similarity to human perception. The method outperforms many existing color reduction algorithms. Third, we propose a novel global orientation map perception using a radial basis function. Finally, we use the derived model along with the brush's position- and force-sensing to produce a visual feedback drawing. Experiments show that our system can generate good paintings including portraits.**

Index Terms – High-level planning, object segmentation, color reduction, orientation map.

I. Introduction

Recently, many areas of research on humanoid robots have been studied, such as motion control, man-machine interfaces, artificial intelligence, and so on. Among them many research projects have tried to create artist robots, with the common objective of exploring new sensing, artificial intelligent (AI), and manipulation techniques. ISAC, [1], for example, is a robot that can track an artist's hand trajectory and mimic it. AARON, [2], is a unique robot that creates a piece of art by itself based on adapting a geometrical model of a subject. [3] considers the possibility of a tele-operated drawing robot that can track markers in the human hand online to generate brush trajectories. Draw-Bot [4] focuses on force feedback to draw a simple pre-programmed shape, whereas [5] considers both visual and force feedback along with grasping technology. Other than artist robots, there are more examples of using robots for specific tasks in order to explore high-level sensing and planning methods, for example, [6] focuses on reasoning techniques of a cook robot, [7] designs a depth perception in a robot that replicates human eye movements.

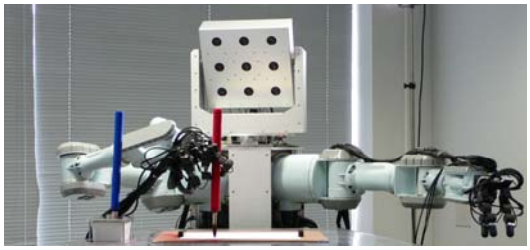


Fig. 1 Humanoid robot painter: Dot-chan.

This paper explores new visual perception techniques, namely object segmentation, color perception, and orientation mapping, and then shows how to apply these methods to high-level manipulation of a painter robot, Dot-chan, shown in Fig.1.

Human painters often concentrate on specific areas of a painting in which details in each area are different. This fundamental observation about the actions of a human painter can be applied to a robot to make generated paintings seem more meaningful. Hence, object segmentation should be used to extract the subject area in a 2D image by letting a human indicate roughly what the subject of a painting is, and letting the robot extract it automatically, like a client and an artist. In object tracking and segmentation, where works like [8] focus on object detection or [9] try to follow an object, these methods cannot return the boundary correctly; hence, this paper focuses on how to segment the object area correctly.

Recent approaches to 2D foreground segmentation attempt to extract the foreground using color information [10], contour information [11], or advanced methods like graph cut [12], [13] which exploit both types of data. However, trading off between color similarity and edge strength is one drawback of graph cut. Furthermore, this method suffers if the foreground/background colors are too ambiguous.

Meanwhile, in the field of object recognition, color distribution-based methods are of limited use, as color alone is usually inadequate to represent patterns. Instead, machine learning dominates this field. Supervised learning techniques such as [14] detect meaningful edges in images by exploiting brightness, color, and texture. They compare the use of density estimation, decision trees, logistic regression, hierarchical mixtures of expert systems, and support vector machines. However, this method cannot extract the foreground directly. [15] also compares various machine-learning algorithms such as neural network, nearest-neighbor, and decision trees, with the objective being to recognize a hand in moving images. The detection result is good even in a changing environment; however, a great deal of user interaction is required during the initial setup. Furthermore, the boundary is not clearly cut comparing to foreground segmentation approaches.

Incorporating the advantages of both foreground segmentation and object recognition, our system augments the decision-making criterion of foreground cut method with a supervised machine-learning algorithm. In addition to each pixel's characteristics being used as training data, gradient information is exploited as a constraint, not as a part of an objective function such as graph cut, to avoid the

trade-off problem. The use of AI improves the detection rate even if foreground and background are very similar.

In addition to the subject's region being segmented, if 3D information is available, geometry edge is derived from multiple view images. Whereas research like [16] or [17] reconstructing a 3D model, this paper uses a method of [5] to process a complex 3D edge into atomic particles that can represent the boundary of an object.

The robot then tries to understand color distribution of the scene to select the best set of colors to use. Conventionally, color reduction uses splitting-based algorithm, such as median-cut [18], or clustering-based algorithm, such as local K-means [19], to restrict the perception of color in a scene. Normally, such methods face the problem that it tends to ignore colors with a small number of pixels. Region-based algorithms are also available, such as [20]. However, none of these methods reflects a human's color perception correctly. This paper proposes a novel color perception method that adapts to the color distribution of each image by incorporating two clustering methods. It is also shown here that our algorithm outperforms Photoshop's perceptual based color reduction [21]. Industrial applications should fall into the area of image compression.

In order to draw brush strokes meaningfully, the robot then senses the orientation of each region. Since orientation data are usually noisy, the global orientation method of [22], which exploits radial basis function [23] to generate an orientation similar in style to Van Gogh, is applied. Our contribution is solving [22]'s weaknesses: loss of subtle gradient information, loss of local control, and the requirement of a great deal of time. This method can be applied for noise reduction in gradient/optical flow-based tracking.

Finally, based on color segments and a planned orientation map, the robot then performs visual feedback drawing. First, it detects a brush and grabs it using cameras and force sensors. Second, a position of the brush tip is calculated using principal component analysis (PCA). Third, it then draws and compares the canvas with the picture in its mind. If not satisfied, using color as an error criterion, it would select a smaller brush and refine the canvas. At the present time, due to limited resources, the robot is programmed to draw only the geometry edge. For color filling, simulation is done on a laptop with 1.8 GHz CPU. The overall time consumed by visual feedback simulation is very low, indicating that it would not be too great a load for the robot to adopt such an algorithm.

These key problems of object segmentation, color perception, orientation map, and visual/force feedback are what our method can directly address. In Section 2, object segmentation using AI is explained. Section 3 explains our novel color perception method. Section 4 demonstrates multi-scale painting using a novel orientation map along with brush detection by camera and force sensors. Finally, Section 5 contains our conclusions.

II. Object Segmentation

In order to extract the subject area in 2D, to draw it in more detail compared with the background, two parties are required to act. A client who asks a robot to draw an image must roughly specify the subject and the background, and the robot would then automatically segment the object. The question is: how would the robot accomplish that? First, instead of letting the robot track a human gesture as a guide, as in [1] or [24], the input is simplified by letting the user mark foreground and background roughly.

As mentioned earlier, a segmentation method that exploits both color and edge information is preferable, such as graph cut. Since the original graph cut algorithm was developed [15], there have been various improvements. [12] focuses on a multidimensional graph cut. [25] improves the graph cut method by allowing interactive estimation and incomplete labeling, both of which reduce the amount of needed user interaction. [26] solves the border problem of [25] automatically, by using belief propagation for matte estimation. The result is quite clean even when the boundary between foreground and background is not smooth. However, this system can take up to 20 minutes to process an input image of 640*480 pixels. Furthermore, trading off between color similarity and edge strength is another drawback of the graph cut method.

On the other hands, the object recognition scheme focuses on exploiting patterns in the image. For example, [15] uses color in RGB domains, hue, and saturation of a large window as the input parameter set of a machine-learning algorithm to distinguish hands in moving images. Interestingly, the method uses 175 input parameters per pixel. So, although the paper asserts the superiority of a neural network, with such large input dimensions one is forced to use a simpler technique, and user interaction is required to specify foreground and background training data. If this is done, the machine can successfully return a segmented image; the user then marks the true negative and false positive areas for use as a new training data. This iterative process must continue until the user is satisfied with the recognition result. In most cases, however, boundary is not correctly segmented.

Our method attempts to exploit the strengths of both of these methods while overcoming their weaknesses.

A. Hybrid object segmentation

Our first core notion is to use AI to distinguish between the object and the background, instead of using a statistical function like GMM or K-means exploited in graph cut. Based on its main strength - effectiveness in recognition - a neural network is the AI basis of this work. For input pattern, the RGB, hue, and saturation data of a pixel are used. To tackle the problem of color ambiguity, variance of a 5x5 gray-scale image is used to represent texture. For topology, a network with one hidden layer is used with the number of hidden nodes set to 30, according to the recommendation of the Levenberg-Marquardt training algorithm used. A sigmoid function is used as a transfer

function so that the output range is between 0 (background) and 1 (foreground)

The second core idea of our method is to optimize region similarity with the constraint of edge strength, avoiding the problem of trading off between both types of data as found in graph cut. The optimization problem can be represented by (1).

$$\min\left\{\sum_{i\in AllPixel} E(i)\right\}; \text{constraint of } C(Grad_i) \tag{1}$$

in which the term that penalizes region similarity is represented by (2).

$$E(i)=\begin{cases}1-\mathrm{AI}_i+C(Grad_i); fore\\ \mathrm{AI}_i+C(Grad_i); back\end{cases} \tag{2}$$

where AI_i is the AI (here, neural network) output of each pixel.

The detailed explanation of our algorithm and more segmentation results can be found in [27].

(a) (b)

Fig. 2 Comparison of segmentation tools. (a) Graph cut implementation of [28], (b) Hybrid segmentation method. The original image is in Fig. 3.

III. Color Perception

After the object boundary is derived, the robot must represent its color using an appropriate set of colors. This section attacks the problem that whether it is possible for a robot to perceive colors in an image in the same way a human does?

The question falls into the domain of color reduction. Many segmentation methods apply splitting [18], clustering [19], or region-based algorithm [20] to restrict the perception of color in a scene. However, none of these methods reflect a human's color perception correctly, as can be seen in Fig. 3.

This paper focuses on a color perception that adapts to color distribution of each image by using maximum distance clustering to initialize K-means clustering. Appropriate color segments would lead to an optimum set of colors that can be used by a painter robot. It is also shown here that our algorithm outperforms Photoshop's proprietary perceptual based color reduction algorithm [21].

A. Clustering based color reduction

Maximum distance clustering (MDC), considered a weak clustering method, is initially used to specify the highest contrasted colors. The set of maximum contrast colors is then used to initialize K-means clustering. K-means clustering is robust and widely used in various fields of research. The drawback is that it tends to ignore colors with a small number of pixels and joins them to a neighboring larger cluster, which lessens the contrast in the output image. Hence, we propose a method that weights the contrasted colors obtained by MDC and the statistically calculated colors obtained by K-means.

To capture the maximum contrast of colors of an image, MDC is applied to the image in the RGB domain. This method starts by sampling a pixel from the input image and then identifying a color with the largest Euclidean distance from the sampled pixel's color. For the second and all consecutive colors, the criterion for a new color is:

$$\min_n \max_{i,j} \left\| c_n - p_{i,j} \right\| \tag{3}$$

where n is the number of clusters, i,j is the coordinate of a pixel, c_n is the mean of each cluster, and $p_{i,j}$ represents RGB colors at each coordinate.

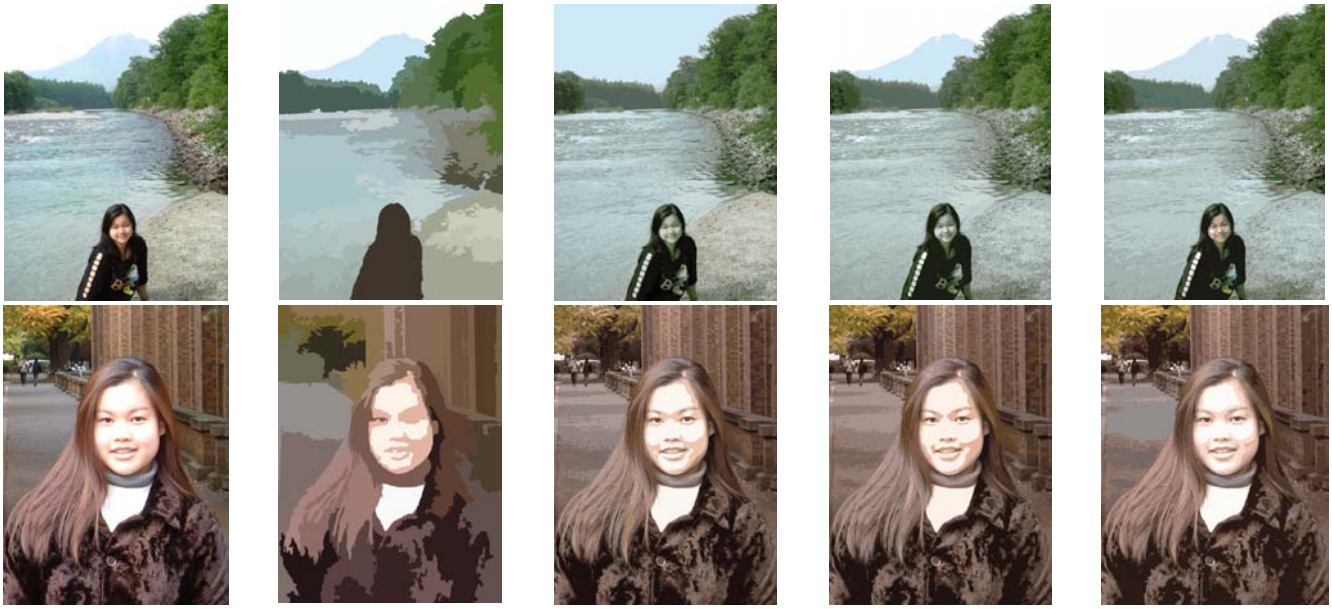
Since the colors would be used to initialize K-means clustering, the algorithm continues until the number of clusters equals the desired number of colors, or there is no other candidate color.

The derived highest contrasted colors are used as initial means for K-means clustering. Then, each new piece of data is used to compute the mean of each cluster from:

$$\min_n \left\| c_n - p_{i,j} \right\| \tag{4}$$

Note that in order to prevent K-means from dominating the clustering process, it must not be run until converged. Practically, running K-means for only one iteration gives the best result.

Finally, each pixel of an image is rendered based on the closest derived cluster mean. As shown in Fig. 3, it can be seen that our method outperforms the region-based algorithm of [20], the commercial perceptual-based color reduction by Photoshop [21], and the converged K-means clustering. On the upper row, the colors on the subject's shirt, skin color, and background colors are well preserved even with the small number of colors. On the lower row, converged K-means is comparable to our method only because the original image contains colors with similar tones, which is not the case for every image. These examples reconfirm our method's efficiency.



(a) Original (b) Wathershed (c) Photoshop (d) K-means (e) MDC+K-means

Fig. 3 Comparison of color reduction tools. (a) Original image, (b) Watershed [20] with 27 colors, (c) Photoshop [21] with 16 colors, (d) Converged K-means with 16 colors, and (e) MDC+K-means with 16 colors.

For a 640x480 image, with n number of colors, running MDC consumes around $0.004*n*n$ seconds whereas K-means clustering consumes only about $0.01*n$ seconds, for each number of colors, on a laptop with 1.8 GHz CPU. The bottleneck of our color reduction is that MDC consumes a lot of time when the number of colors is high.

IV. Visual Feedback Drawing

After an appropriate number of colors is derived, a method that mimics human painting style is proposed. The first contribution is a method that can analyze orientation in an image the same way as humans do. Second, our research also attempts to verify the visual feedback, to see how a robot exploits brushes and a limited number of colors in hands to generate "artwork."

For the local information used to guide brush strokes, [29] uses a gradient to guide brush strokes that are more robust to texture, whereas [30] uses image moment that is more robust to area orientation. However, such methods do not produce good brush strokes because the gradient or moment that a robot preserves could be noisy. Recently, methods that consider global gradient include [22], which selects only a strong gradient and applies radial basis function (RBF) to interpolate a gradient on other areas. This seems to match an artist's perception. In fact, results shown in [22] resemble Van Gogh's style. Our work modifies the method of [22] to generate practical end-effector trajectories for the robot.

In addition to orientation, visual feedback is divided into hierarchies depending on the brush size the robot has, in which the detail is proportional to the brush size. The largest brush is used first and smaller ones are used to add details and solve any imperfections introduced by prior hierarchies. For low-level manipulation, please refer to [5].

A. Orientation perception

From an original image, the robot would then calculate the normal orientation. Orientation, (5), is calculated using the gradient operator.

$$\theta = \tan^{-1}(Grad_y / Grad_x) \tag{5}$$

, and θ has a range between $\{-\pi/2, \pi/2\}$.

This normal vector is not directly used to guide the brush because it would be noisy as shown in Fig. 4. Instead, the robot uses linear basis RBF [23] to interpolate the gradient field. Differing from [22], where only strong edges are used as an input for RBF, and which could fail to capture some subtle but important edge such as human hair, this research exploits all gradient values as an input for RBF, which is possible by weighting function of (6).

$$w = \sqrt{Grad_x^{\,2} + Grad_y^{\,2}} \tag{6}$$

Furthermore, the orientation image is not calculated totally globally since doing so would make an orientation of any pixel affected by the whole image's orientation, making straight lines bend. Also, globally computing consumes a lot of time; for each pixel, the number of operations required is shown in (7).

$$R*C \tag{7}$$

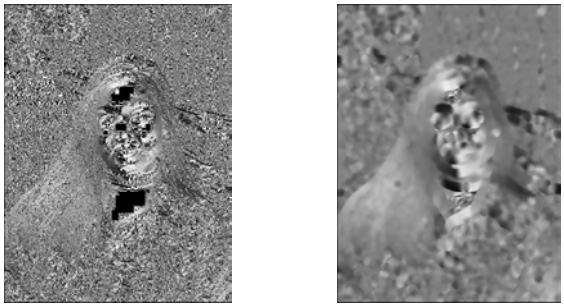
where R is the number of rows and C is the number of columns of an image. For example, a 640x480 image would require overall 9.4372e+010 sets of operations.

In order to enable the robot to calculate this in a reasonable time, instead, the robot just focuses around an area and calculates global orientation in this area. By doing this, the processing time would be reduced as shown in (8).

$$r*c \tag{8}$$

where r is the number of rows and c is the number of columns of a mask. For example, a 640x480 image with a 10x10 mask, used in this work, would require only around 3.0720e+007 sets of operations. In the case that there is no

information of input orientation in the mask, as can be seen from the input figure, the mask size for searching is doubled until orientation information is found. In other words, the area that has no orientation information would use the orientation of the surrounding area. The result is shown in Fig. 4.



(a) (b)

Fig. 4 Orientation of an image scaled from $\{-\pi/2, \pi/2\}$ to $\{0, 255\}$. (a) Original orientation of each pixel, (b) Orientation after RBF is applied.

As can be clearly seen on the subject hair and on the building, the result is very much the same as when humans perceive orientation in a scene, even with the presence of a noisy gradient.

B. Visual feedback drawing: plan, act, and sense

First, the robot detects a brush using stereo cameras mounted in its head as can be seen in Fig. 1. Whereas a former grasping working like [31] focuses on localization of an object and path planning, or [32] focuses on learning from observation, the low level control of this work considers localization and grasping using multi-sensors: cameras and force sensors in the robot's hand.

After picking up the brush, in order to calibrate the position of the brush tip, based on a real-time captured image, principal component analysis (PCA) is used to calculate the axis of the brush in its hand. Based on this axis, pre-computed distance between the lower edge of the handle and the tip is added to the lowest position of the brush handle detected. This is established as the brush tip position.

The robot then scans each color segment derived from section 3 to find an area that contains a number of pixels larger than (9).

$$th\_region = k\_region \times r^2 \quad (9)$$

where r is the pre-computed radius of the brush, and k_region is the constant that must be lower than π, depending on drawing style.

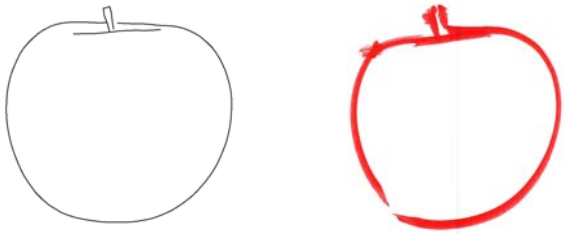
If such an area is found, the robot starts to draw. To check whether the brush tip touches the canvas or not, the force sensor is then used along with the position of the brush tip detected in real time.

For the next movement, the robot then focuses along the normal direction of orientation derived earlier, and counts all yet-to-be-drawn pixels with the same color. If the number of pixels is higher than the threshold described in (9), the robot would decide to move the brush.

Each stroke is considered finished if an area found in the normal orientation direction is smaller than (9). After all the colors in each hierarchy are painted, the next hierarchy starts, where the robot then select a smaller brush. The number of hierarchies depends on the error criterion between the picture in the robot's mind and the painting on the canvas.

Finally, in a 3D case where the geometry edge can be derived, using the method of [5], an edge can be drawn using the smallest brush size available.

At the present time, due to insufficient drawing equipment and other resources, the artworks the robot created are subjected only to drawing of the geometry edge, as can be seen in Fig. 5. In color filling, simulation on a laptop is done, *k_region* is set to be 0.5 for the first iteration and 0.3 for consecutive iterations, and brush size is reduced by the ratio of 0.5 every hierarchy. Also, prior object segmentation allows for the background to usually require less detail than the foreground, so it can be drawn with a smaller number of hierarchies. This usually makes the foreground stand out and also leads to a lot of time reduction, an advantage that can be seen in Fig. 6.



(a) (b)

Fig. 5 Geometry edge drawn by robot. (a) Input processed by algorithm of [5], and (b) Output.

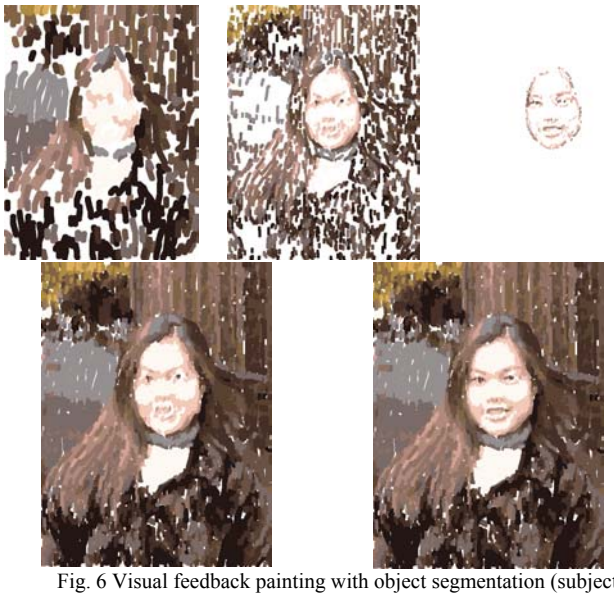


Fig. 6 Visual feedback painting with object segmentation (subject's face here). The first row shows each hierarchy, and the second row shows the overall result.

This type of visual feedback simulation consumes around 1 second per hierarchy, for a 640x480 image, on a laptop with 1.8GHz CPU; thus, it should not be any load at

all for the real robot platform. The problem to be aware of in the real drawing by the robot is how to consider the color mixing on canvas, and how it would affect the drawing as a whole.

V. Conclusion

This paper focuses on finding new robot vision technologies of object segmentation, color perception, and orientation mapping, and applies them to accomplish visual feedback drawing by a robot.

In order to extract the subject area, a client roughly specifies the object and the background, and the robot then automatically segments the object. A hybrid object segmentation that exploits a novel optimization technique and machine learning is used. Instead of considering the quality of boundary data as a parameter in an optimization as in graph cut, it is considered as a constraint to temporarily stop the growing process. The powerful machine learning technique of a neural network is used to recognize an input image's color and texture characteristics. The neural network does not pre-assume the data distribution, and thus is flexible and robust. The present problem of this method is that is time consuming.

Resulting in similar color perception to a human, maximum distance clustering and K-means clustering-based color reduction are used for the robot to sense color distribution in a scene. The method outperforms many existing color reduction algorithms in the aspect of color perception. The drawback is that maximum distance clustering consumes a lot of time when the number of colors is high.

A fast but delicate global orientation perception is proposed using a radial basis function. The improvement over a previous orientation calculation method that uses RBF is higher sensitivity to pixels with low amplitude of gradient, local controllability, and time reduction.

Visual feedback painting then uses derived color segments and an orientation map to guide brush strokes hierarchically. Cameras on the robot's head and force sensors in its hand are used to locate and grasp the brush as well as to check whether or not the brush contacts the canvas.

Acknowledgment

This work is supported in part by the Japan Science and Technology Corporation (JST) under the CREST project.

References

[1] A. Srikaew, M. E. Cambron, S. Northrup, R. A. Peters II, M. Wilkes, and K. Kawamura, "Humanoid Drawing Robot," *IASTED International Conference on Robotics and Manufacturing*, 1998

[2] P. Monaghan, "An Art Professor Uses Artificial Intelligence to Create a Computer That Can Draws and Paints," *The Chronicle of Higher Education*, 1997.

[3] http://techhouse.brown.edu/~neel/drawing_telerobot/

[4] http://robix.com/drawbot.htm

[5] S. Kudoh, K. Ogawara, M. Ruchanurucks, and K. Ikeuchi, "Painting Robot with Multi-Fingered Hands and Stereo Vision," *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, 2006.

[6] F. Gravot, A. Haneda, K. Okada, and M. Inab, "Cooking for Humanoid Robot, a Task that Needs Symbolic and Geometric Reasonings," *ICRA*, 2006.

[7] F. Santini and M. Rucci, "Depth Perception in an Anthropomorphic Robot that Replicates Human Eye Movements," *ICRA*, 2006.

[8] D. R. Thompson and D. Wettergreen, "Multi-object Detection in Natural Scenes with Multiple-view EM Clustering," *IROS,* 2005.

[9] M. Marrn, J. C. Garca, M. A. Sotelo, D. Fernndez, and D. Pizarro, "XPFCP: An Extended Particle Filter for Tracking Multiple and Dynamic Objects in Complex Environment," *IROS,* 2005.

[10] Y. Y. Chuang, B. Curless, D. Salesin, and R. Szeliski, "A Bayesian approach to digital matting," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2001.

[11] E. Mortensen and W. Barrett, "Intelligent scissors for image composition," *ACM SIGGRAPH*, 1995.

[12] Y. Boykov and M. P. Jolly, "Interactive Graph Cuts for Optimal Boundary and Region Segmentation of Objects in N-D Images," *IEEE Internation Conference on Computer Vision*, 2001.

[13] D. Greig, B. Porteous, and A. Seheult, "Exact MAP Estimation for Binary Images," *J. Roy. Stat. Soc. B. 51*, 1989, pg. 271–279.

[14] D. R. Martin, C. C. Fowlkes, and J. Malik, "Learning to Detect Natural Image Boundaries Using Local Brightness, Color, and Texture Cues," *IEEE Trans. on Pattern Analysis and Machine Intelligence 26*, 2004.

[15] J. A. Fails and D. R. Olsen, "A Design Tool for Camera-Based Interaction," *Conference on Human Factors in Computing Systems*, 2003.

[16] Y. Iwashita, R. Kurazume, K. Hara, and T. Hasegawa, "Robust Motion Capture System against Target Occlusion using Fast Level Set Method," *ICRA*, 2006.

[17] M. Tomono, "Dense Object Modeling for 3-D Map Building Using Segment-Based Surface Interpolation," *ICRA*, 2006.

[18] P. Heckbert, "Color image quantization for frame buffer display," *Comput. Graph.*, vol. 16, pp. 297–307, 1982.

[19] O. Verevka, "The local K-means algorithm for color image quantization," M.Sc. dissertation, Univ. Alberta, Canada, 1995.

[20] L. Vincent and P. Soille, "Watersheds in Digital Spaces: An Efficient Algorithm Based on Immersion Simulations," *IEEE Trans. on Pattern Analysis and Machine Intelligence 13*, 1991.

[21] Adobe Photoshop 7.

[22] J. Hays and I. Essa, "Image and Video Based Painterly Animation," *International Symposium on Non-Photorealistic Animation and Rendering*, 2004.

[23] A. G. Bors, "Introduction of Radial Basis Function (RBF) Networks," *Online Symposium for Electronics Engineers*, vol.1 of *DSP Algorithms: Multimedia*.

[24] C. Breazeal, C. Kidd, A. L. Thomaz, G. Hoffman, and M. Berlin, "Effects of Nonverbal Communication on Efficiency and Robustness in Human-Robot Teamwork," *IROS,* 2005.

[25] C. Rother, V. Kolmogorov, and A. Blake, "GrabCut – Interactive Foreground Extraction Using Iterated Graph Cut," *ACM SIGGRAPH*, 2004.

[26] J. Wang and M. F. Cohen, "An Iterative Method Approach for Unified Image Segmentation and Matting," *IEEE International Conference on Computer Vision*, 2005.

[27] M. Ruchanurucks, K. Ogawara, and K. Ikeuchi, "Neural Network Based Foreground Segmentation with an Application to Multi-Sensor 3D Modeling," *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, 2006.

[28] Y. Li, J. Sun, C. K. Tang, and H. Y. Shum, "Lazy snapping," *ACM SIGGRAPH*, 2004.

[29] A. Hertzmann, "Painterly Rendering with Curved Brush Strokes of Multiple Sizes," *ACM SIGGRAPH* , 1998.

[30] M. Shiraishi and Y. Yamaguchi, "An Algorithm for Automatic Painterly Rendering Based on Local Source Image Approximation," *International Symposium on Non-Photorealistic Animation and Rendering*, 2000.

[31] Y. Hirano, K. Kitahama, and S. Yoshizawa, "Image-Based Object Recognition and Dexterous Hand/Arm Motion Planning Using RRTs for Grasping in Cluttered Scene," *IROS,* 2005.

[32] M. Hueser, T. Baier, and J. Zhang, "Learning Demonstrated Grasping Skills by Stereoscopic Tracking of Human Hand Recognition," *ICRA*, 2006.