The 2010 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 18-22, 2010, Taipei, Taiwan

SIFT-Cloud-Model for Object Detection and Pose Estimation with GPGPU Acceleration

Takahiro Nakada[1,2], Satoshi Kagami[1,2,3], Hiroshi Mizoguchi[2,1]

Abstract— **A function to find objects and to estimate their poses is crucial for a home service robot that works in human living environment.**

In this paper, we propose a "SIFT-Cloud-Model (SCM)", that is designed to represent objects in daily scene. The model consists of SIFT feature descriptors, and their 3D positions, and eye vectors. The model is automatically obtained by using Structure from Motion techniques. Then it is matched against single camera images by finding the corresponding SIFT features, and using them to estimate the object pose.

The system is optimized using GPGPU methods. Finally experimental results for accuracy and calculation time are shown. The method achieves 5.5 fps with 8% error in translation and 25[deg] error in rotation.

I. Introduction

A vision for robots is that they will assist or help humans in daily life by acting as a substitute human to do physical tasks. One ot the difficulties in achieving this task is perception. Robots needs to have a perception function to find out target objects in a complex daily scene, and to estimate relative pose for manipulation. If such a perception function can be achieved, there will be many possible applications.

So far, many model based methods for finding target object, and for estimating relative pose, were proposed in computer vision field.

These methods can be categorized into three groups, the first is 2-D image based (ex. [1][2][3]), the second is 3-D shape model based (ex. [4][5]), and the third is texture mapped 3-D shape model based (ex.[6]). 2-D image based object finding has difficulty in estimating arbitrary rotations of the target object. 3-D shape models have difficulty matching against partial views obtained from robot sensors. Texture mapped 3-D shape models have a problems with accuracy and computational cost.

The proposed method is a modification of the third method. Instead of having a texture mapped 3-D shape model, it has SIFT descriptors with 3-D position and camera view vectors. This model named "SIFT-Cloud-Model(SCM)" and it is designed to be accurately and efficiently matched with an input image. This model is easily implemented by using General-Purpose Computation on Graphics Hardware(GPGPU) techniques[7].

1: Digital Human Research Center, National Institute of Advanced Industrial Science and Technology, 2-3-26, Aomi, Koto-ku, Tokyo, 135-0064, Japan.
2: Dept. of Mechanical Engineering, Tokyo University of Science
3: Japan Science and Technology Agency (JST)

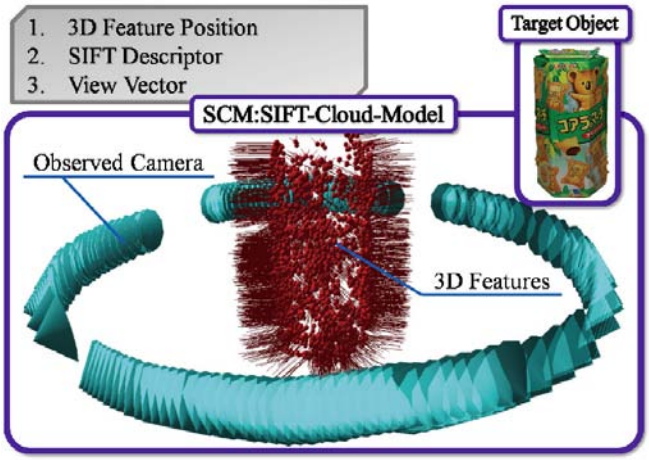


Fig. 1. SIFT-Cloud-Model

II. SIFT-CLOUD-MODEL (SCM)

A. Model Description

The SCM formulated in this paper is composed of the following three types of information:

1. 3-D feature position X_i'
2. SIFT descriptor[8] $descr_i'$
3. View vector $M_i'\,[R_i'|T_i']$

Here, scalar or vector with single quote $'$ means that is in SCM model throughout this paper (ex. X_i').

Fig. 1 center shows dots that represent 1) and 2). Bars that start from those dots and cones show the estimated camera positions that correspond to 3).

SIFT is now widely used to represent features in a scene and it is rotation, scale and lighting invariant in thory. In practice, there is a limit for changes in scale and lighting. A SIFT descriptors is a 128 dimensional vector converted from intensity gradient around the feature.

The SIFT descriptor itself is a 2-D feature of an image, captured from a certain location from which the target object appears in given position and orientation. According to the viewpoint, the appearance of every point in 3D surface may change. Therefore one SIFT descriptor is not enough to represent one surface point of target object in 3-D space. We add 3-D positions of the sift features X_i' and the camera view vectors frmo where the feature was observed $M_i'\,[R_i'|T_i']$, in order to describe 3-D target objects.

B. Model Generation

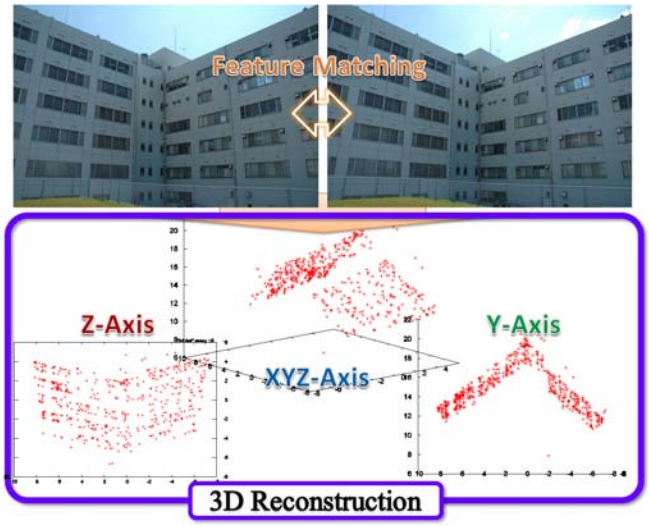


Fig. 2. Structure from Motion of Two Consequtive Images

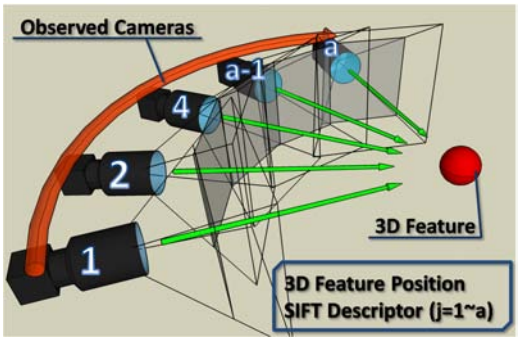


Fig. 3. SIFT descriptors from multiple Observation Camera Locations

1) 3D Feature: In order to obtain 3-D feature positions, Structure from Motion technique[9] is adopted. At first, corresponding features are obtained from two consecutive images by matching SIFT vectors. Then fundamental matrix of these corresponding points is calculated using SFM. This fundamental matrix is decomposed into intrinsic and external camera parameters. Finally, 3-D reconstruction of those corresponding points are calculated using these parameters by triangulation. Fig. 2 shows reconstructed 3-D feature points from these two images.

2) Selection of Representative View Vectors: SFM reconstructs not only 3-D feature points but also camera locations at the same time. Each feature point in 3-D can be seen from many views with slightly different SIFT descriptors. Fig. 3 shows this relationship. In order to reduce computational cost, we would like to reduce this information into one representative SIFT descriptor with one view vector at each 3-D feature point.

In Fig. 3, there are SIFT descriptors ($descr_1$, $descr_2$, $\cdots$, $descr_j$, $\cdots$,$descr_a$) that correspond to each view vector that is looking at the same 3-D point. Each SIFT descriptor is obtained from a different camera location. Equation (1) is used to calculate the distance between corresponding features

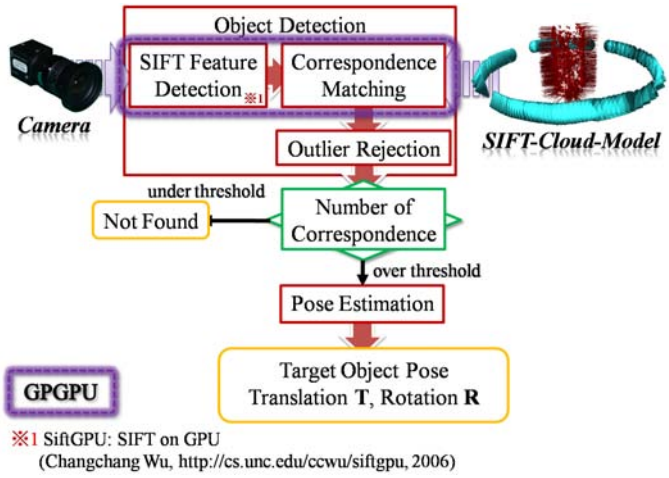


Fig. 4. Overview of Object Detection and Pose Estimation using SIFT-Cloud-Model

observed from different views. The SIFT descriptor that is in the center of the group of corresponding features is determined to be the representative SIFT feature $descr'_i$, and its view vector is determined to be the representative view vector $M'_i\,[R'_i|T'_i]$.

$$\sum_{r=1}^{128}(descr_{k,r}-descr_{j,r})^2 < threshold \tag{1}$$

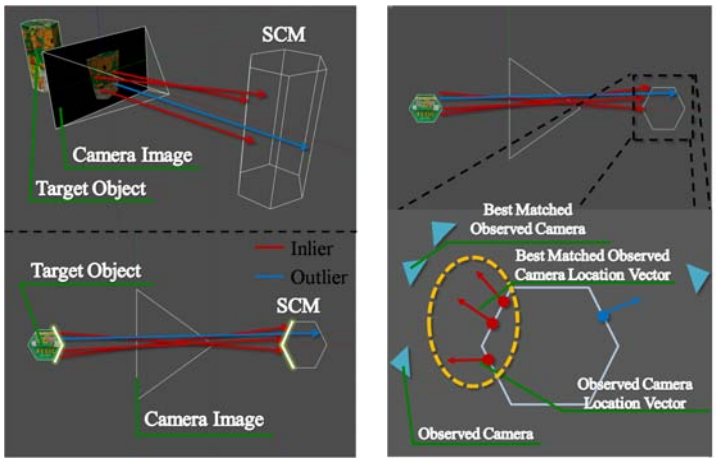
III. Detection of Target Object using SCM

Building SCM is a offline process. Next, detection and pose estimation using the SCM needs to be done online. Fig. 4 summarizes our proposed detection and pose estimation method. At first, 2-D SIFT features are calculated from camera input image. Then this 2-D SIFT features are matched against SCM features together with camera location of SCM model. Next, it rejects outliers among those corresponding points by extending Hess's method[10]. The original method is for outlier rejection in 2D-2D corresponding points using a perspective transform matrix. Therefore in 2D-3D correspondence problem here, we introduce View Vector in SCM in order to reject outliers. Finally, the target object is detected in camera image using 2D-3D corresponding points matched above. Position and orientation of the target object related to the camera is obtained from 2D-3D correspondence of feature points by using the POSIT(Pose from Orthography and Scaling with ITerations) method[11].

SIFT feature calculation and its correspondence matching will be calculated in GPGPU technique. As for SIFT feature calculation, Wu's implementation is adopted.

A. Correspondence Matching

SIFT feature is designed to be rotation, scale, and intensity invariant, but it is not designed to be affine invariant. So a SIFT feature that was detected from one point on a surface, cannot, in practice, be autocorrelated with changes in view direction of more than 30[deg]. Therefore, we added the view vectors to the SCM, in order to help matching against



(a) Feature Matching using SIFT Descriptor Only

(b) Select Inlier using Camera View Vector

Fig. 5. Feature Matching using SIFT Descriptor and View Vector

camera input image. If correspondence is made by only using SIFT features in between camera image and in SCM, it may contain mismatches from matches against the wrong side of the target object. Fig. 5(a) shows this example.

Instead, by using view vectors $M_i'[R_i'|T_i']$ in the SCM model, correspondence of 2-D image with 3-D SCM model can be made by omitting mismatches, and simultaneously the correspondence of the view vectors $M_{opt}[R_{opt}|T_{opt}]$ are found. Fig. 5(b) shows an example of this.

B. Outlier Rejection

The method shown in previous section rejects mismatch by using geometric constraints. However there are also mismatches caused by SIFT descriptor correspondence.

Hess *et al.* proposed outlier rejection method in 2D-2D correspondence[10]. Their method randomly selects four corresponding points $\mathbf{x}_j(x_j,y_j,1), \mathbf{x}_j'(x_j',y_j',1)(j=1\sim 4)$, and generates a matrix $\mathbf{H}$ that satisfies Equation (2). Here (x_j,y_j) means a feature in camera image, and (x_j',y_j') means corresponding feature on SCM that was projected in 2-D. By iteratively omitting those points that satisfy (3) using this matrix $\mathbf{H}$, the method finds out a matrix $\mathbf{H}$ and outlier that gives the minimum number of rejections.

$$s\mathbf{x}_j = \mathbf{H}\mathbf{x}_j'^T \tag{2}$$

$$d(\mathbf{x}_i, \mathbf{H}\mathbf{x}_i') > threshold \tag{3}$$

However, Hess's method can't directly apply to the case of 3D-2D correspondance. In our case, using the view vector obtained in Section III-A, SCM 3D feature points are projected onto 2D features. This is done by projecting 3-D feature point X_i by using the best matched observation camera location $M_{opt}[R_{opt}|T_{opt}]$ with Equation (4). Matrix K in Equation (4) are the intrinsic parameters of the camera. This procedure converts the 3D-2D correspondence problem into A 2D-2D correspondence problem, and hence we can apply Hess's method. Fig. 6 shows this process.

$$s\mathbf{x}_i' = KM_{opt}\mathbf{X}_i' \tag{4}$$

IV. Pose Estimation of Target Object

After the previous section, the feature points in 2-D and the corresponding points in 3-D SCM model are obtained. Then the POSIT algorithm proposed by DeMenthon[11] is adopted to robustly estimate a target pose.

Fig. 7 shows a pin-hole camera model with an optical center O, focal length f and image plane G. Let the coordinates of this camera be $(\mathbf{i},\mathbf{j},\mathbf{k})$. In this image space, 3-D feature points $\mathbf{X}_i'$ and object coordinates $(\mathbf{u},\mathbf{v},\mathbf{w})$ with X_0 as origin exist. POSIT calculates the target pose $M[R|T]$ by Equation (5).

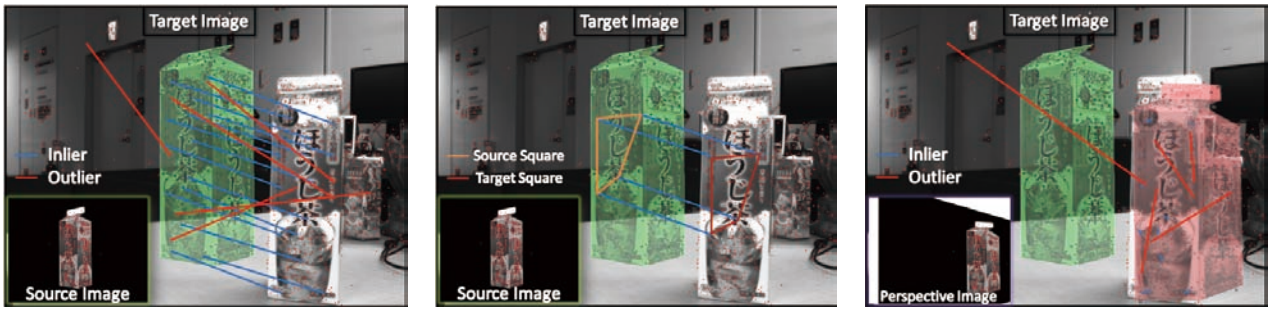
$$\mathbf{T} = \mathbf{X}_0' = \begin{pmatrix} X_0' \\ Y_0' \\ Z_0' \end{pmatrix} \quad \mathbf{R} = \begin{bmatrix} i_u & i_v & i_w \\ j_u & j_v & j_w \\ k_u & k_v & k_w \end{bmatrix} \tag{5}$$

A. Determination of The Estimation Outcome

POSIT iteratively calculates the statistical pose of a target from the 2-D projection of the 3-D model. When the number of iterations increase, the result becomes more accurate. However, the method can't guarantee a quality of its result. Therefore, we need to confirm obtained result in some way.

In this paper, we applied two methods. One is by comparing the current camera view vector with the model's SIFT descriptor view vector $M_{opt}[R_{opt}|T_{opt}]$, which is mentioned in Section III-A.

The other is by using the reprojection error of obtained POSIT result. The POSIT result is used to reproject the 3-D target object into 2-D features by Equation (6). Here x_i''



(a) Correspondence Contains Outlier

(b) Perspective Projection Transform of Four Points

(c) Outlier Rejection

Fig. 6. Outlier Rejection using Perspective Projection Matrix

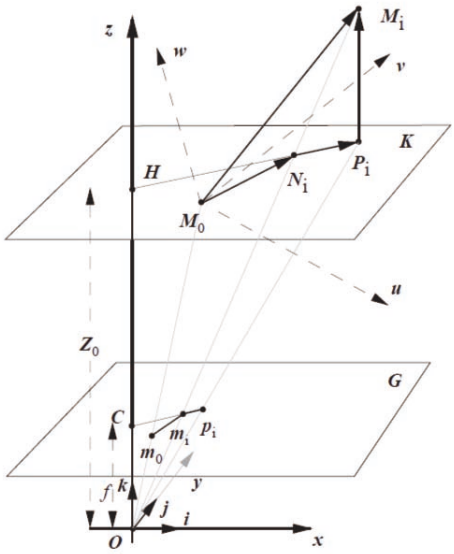


Fig. 7. Perspective projection and scaled orthographic projection for an object point and a reference point

means the reprojected feature position in 2-D image. Then the 2-D distance of those corresponding features gives an error value of pose estimation result. There is a threshold to cut off the error.

$$s\mathbf{x}_i'' = \mathbf{KMX}_i' \tag{6}$$

$$\sum^{n} d(\mathbf{x}_i, \mathbf{x}_i'') < threshold \tag{7}$$

V. Experiments

Detection ratio and pose estimation accuracy is measured using proposed SCM. Computational speed up using GPGPU is also measured.

As for experiments, a firewire (1394b) camera Point Grey Research FL2-08S2C(Table I), megapixel lens Spacecom HHF4MK-3C(Table II), PC(Table III) and graphics hardware NVIDIA Geforce GTX280(Table IV) for GPGPU are used.

TABLE I
Camera FL2-08S2C (Point Grey Research)

Image Sensor Model	Sony ICX204 1/3"
Resolution	1024x768
Pixel Size	4.65 x 4.65 μ m
Image Data Formats	YUV422
Frame Rate	30 fps

TABLE II
Lens HHF4MK-3C (SPACECOM)

Focal Length	4 mm
Maximum Relative Aperture	1:1.8
Iris	F1.8 - 16
Angular Field of View	83.4x64.5 deg.
Minimum Focus Distance	0.1 m

A. Object Detection Performance

In order to evaluate detection performance of the proposed model, an experiment was conducted by occluding target objects. Fig. 8 shows the experimental setup. Detection rate of the target object was measured by changing occluded rate and number of SIFT feature points. Occlusion is done by placing other object in front of target object.

Fig. 9 shows the detection rate with 2,700 SIFT features in an input image. In this experiment, even if 60% of the target object was covered, detection rate was higher than 90%. In 100% occluded case (that means any part of the target doesn't be seen), there is no false detection.

TABLE III
PC specifications

OS	Ubuntu9.10 Kernel 2.6.31-17
CPU	Intel Core2Duo E8400 3.00 GHz
Memory	DDR3 2.0 GB
Compiler	GCC 4.1
GPGPU	NVIDIA Cg Toolkit 2.1 CUDA 2.3 & CUDA SDK 2.3

TABLE IV
GEFORCE GTX280 SPECIFICATIONS

Stream Processors	240
Graphics Clock	602 MHz
Processor Clock	1296 MHz
Texture Fill Rate	48.2 billion/sec
Memory Clock	1107 MHz
Standard Memory Config	1024 MB

Fig. 8. Experimental Setup of Target Object Detection with Occlusion

Fig. 10 shows the detection ratio result by changing the amount of occlusion and the number of features detected. With more than 1,000 features and less than 80% occlusion, the target object is robustly detected.

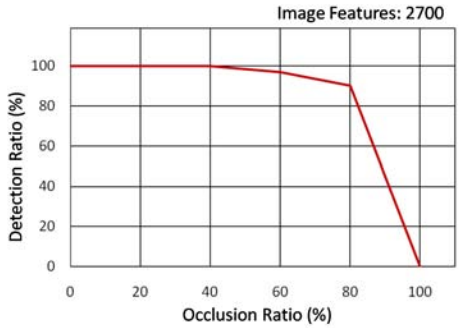


Fig. 9. Object Detection Ratio of Camera Image with 2700 SIFT Features in Input Image

B. Pose Estimation Accuracy

We put the target object in the same scene but with no occlusion by the other objects, and measured translational and rotational accuracy.

Fig. 11 shows translational error related to the distance from the camera. Translational accuracy was about 7% related to the distance. Rotational accuracy in roll angle related to the true value is shown in Fig. 12. Rotational accuracy in roll angle was about 24[deg] throughout 360[deg] rotation.

C. Computation Time

In order to confirm speed up of computation time by using GPGPU, comparison in between CPU (CPU based) and

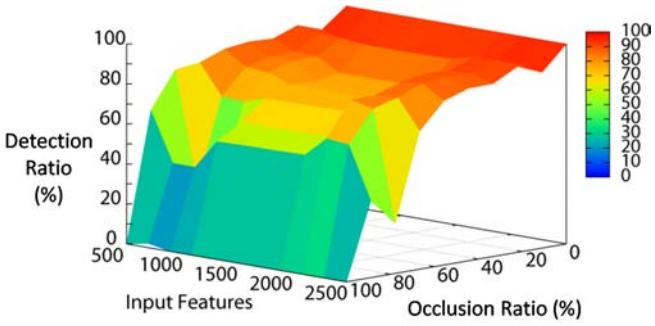


Fig. 10. Relationship in between Number of Input Features, Occlusion and Detection ratio

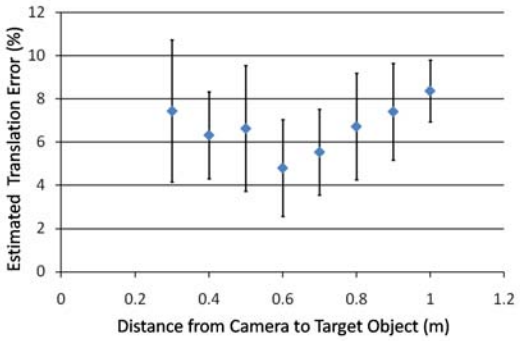


Fig. 11. Pose estimated error of translation

CPU combined with GPGPU (GPU based) were conducted. GPGPU acceleration is mainly used for the correspondence calculation of SIFT features that has length of 128 vector. For SIFT detection by CPU, Lowe's implemention[12] was used, and for GPU, Wu's implementation[13] was used.

As for the correspondence calculation, several methods were proposed, and we adopted Nearest Neighbor Search(NNS)[14] for GPGPU purpose. NNS is trying to compare all possible combinations with a threshold based cut off. This method basically guarantees to find out the best match. However NNS is computationally very expensive for

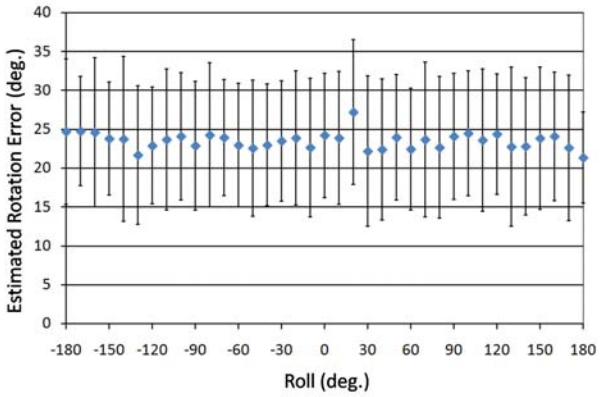


Fig. 12. Pose estimated error of rotation

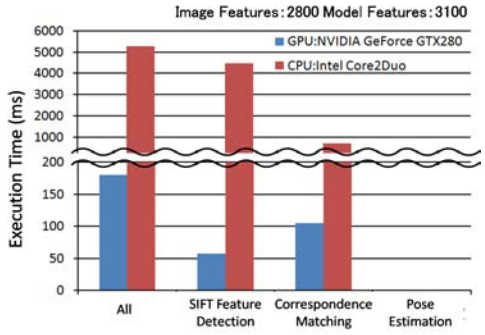
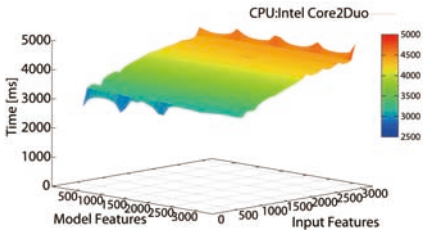


Fig. 13. Execution time: image features 2800, model features 3100

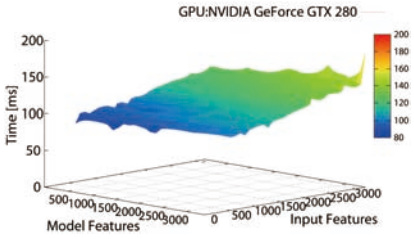
CPU, so the Best Bin First Search (BBF) method[15] was adopted for CPU. The method preprocesses given data (in this case SCM) and in matching phase it quickly finds out an approximate candidate.

Fig. 13 shows computational time for CPU based and GPU based implementation with 2,800 SIFT features in input camera image, and 3,100 SIFT features in SCM. According to this figure, about 98% of execution time was spent for SIFT feature detection and correspondence matching. Using GPU, SIFT feature detection becomes 78 times faster, correspondence matching becomes 7 times faster, and in total it becomes about 30 times faster than CPU. Cycle rate in this condition was about 5.5 fps, thus the method can be used while robot is in motion.

Fig. 14(a) shows the relationship between the number of features and calculation time in CPU, and Fig. 14(b) shows the same for GPU. From this result, GPU based implementation is not sensitive to the number of input features nor the number of SCM features.



(a) CPU Based



(b) GPU Based

Fig. 14. Execution time

VI. Conclusion

This paper proposed a 3-D object description method "SIFT-Cloud Model" that consists of SIFT feature vectors, their 3-D positions and view vectors. The model is designed to be computationally efficient and to detect and estimate relative pose from an input camera image. Detection and pose estimation methods can be implemented using thee GPGPU approach, and from the experimental results, current hardware reached 5.5 fps with 1024x768 input image size.

Experimental results shows that the proposed method detects a given object from the input camera image even if occlusion occurs. The method is robust to background image features, and it can detect target objects that are occluded up to 60% with more than a 90% detection ratio. The method can be used to detect objects found in daily scenes in real-time. With this system, a treasure hunting application for a mobile robot becomes possible.

References

[1] J.P. Lewis, "Fast Template Matching", *Vision Interface*, vol. 120, 1995, pp 123

[2] J.A. Richards and X. Jia, "Histogram Matching", *Remote Sensing Digital Image Analysis*, Springer Berlin, chapter 4, 1993, pp 97-100

[3] R. Lienhart and J. Maydt, "An Extended Set of Haar-Like Features for Rapid Object Detection", *IEEE International Conference on Image Processing*, vol. 1, 2002, pp 900-903

[4] K. Kanatani and K. Yamada K. Kanatani and K. Yamada, "Model-Based Determination of Object Position and Orientation without Matching", *Journal of Information Processing*, vol. 12, No. 1, 1989, pp 1-8

[5] P.J. Besl and H.D. McKay, "A Method for Registration of 3-D Shapes", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, No. 2, 1992, pp 239-256

[6] H. Yaguchi and K. Okada and M. Inaba, "Model matching and 3D Pose Estimation Method using High-speeded Texture Image Search", *The 27Th Annual Conference of the Robotics Society of Japan*, 2009

[7] D. Luebke and M. Harris and N. Govindaraju and A. Lefohn and M. Houston and J. Owens and M. Segal and M. Papakipos and I. Buck, "GPGPU: General-Purpose Computation on Graphics Hardware", *The 2006 IEEE Association for Computing Machinery*, 2006

[8] D.G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints", *International Journal of Computer Vision*, vol 60, No. 2, 2004, pp 91-110

[9] R. Hartley and A. Zisserman, "Multiple View Geometry in Computer Vision", *Cambridge University Press*, chapter. Structure from Motion, 2003

[10] R. Hess, "Sift Feature Detector", *http://web.engr.oregonstate.edu/hess/*, 2009

[11] D.F. DeMenthon and L.S. Davis, "Model-based Object Pose in 25 Lines of Code", *International Journal of Computer Vision*, vol 15, No. 15, 1995, pp 123-141

[12] D. Lowe, "Demo Software: SIFT Keypoint Detector", http://www.cs.ubc.ca/ lowe/keypoints/

[13] C. Wu, "SiftGPU: A GPU Implementation of Scale Invariant Feature Transform (SIFT)", http://cs.unc.edu/ ccwu/siftgpu/, 2006-2009

[14] K. Beyer and J. Goldstein and R. Ramakrishnan and U. Shaft, "When is" nearest neighbor" meaningful?", *Lecture Notes in Computer Science*, 1999, pp 217-235

[15] J. Beis and D. Lowe, "Shape indexing using approximate nearest-neighbour search in high-dimensional spaces", *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1997, pp 1000-1006