Proceedings of the Fifth IIEEJ International Workshop
on Image Electronics and Visual Computing 2017
Da Nang, Vietnam, February 28- March 3, 2017

IMPROVED KEYPOINT PATCHES IN EVOLUTIONARY POINT CLOUDS REGISTRATION

Kazuma Uenishi† Jaime Sandoval† Munetoshi Iwakiri†† Kiyoshi Tanaka†
†Shinshu University ††National Defense Academy of Japan

ABSTRACT

Registration is the process of aligning 3D point clouds. It is frequently applied to many tasks such as 3D modeling and object recognition. Unlike most registration methods, evolutionary computation registration has the advantage to work without depending on the initial position of the point clouds. However, its huge computation cost is its major issue. To solve this, we proposed a novel approach that computes the fitness only by using the characteristic parts of one point cloud. A KeyPoint Patch (KPP) is a point cloud extracted around a keypoint, when using KPPs for registration, it has shorter processing time, and in many cases equivalent registration accuracy than other evolutionary approaches. On the other hand, in previous works we confirmed that there are cases in which it always converge to a certain local solution, moreover, the correlation between the processing time and registration accuracy is not evaluated. In this paper, we proposed an accelerated KPP registration that escapes from local solutions and evaluated its performance. The former method separated positions of each KPP farther from the boundaries to reduce the negative influence of occlusion. This method attempts to further speed up the registration by reducing the number of keypoint patches and its points. From the experiments results, we confirmed that the proposed method is effective and is almost 2.6 times faster than previous methods.

1. INTRODUCTION

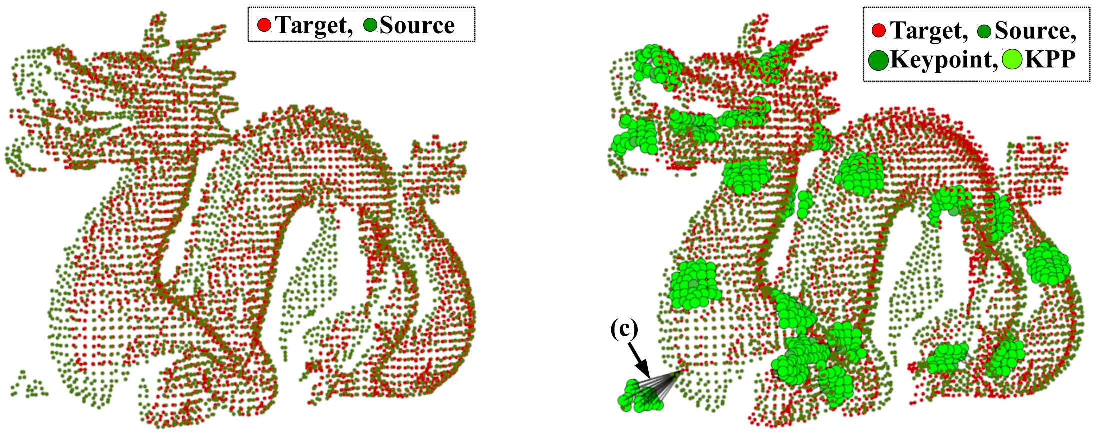
As 3D sensors are becoming popular, 3D point cloud processing is also becoming an important research field in computer vision. A 3D point cloud is a data structure that represents the shape of the surface of objects by numerous points. However, there are many problems in point clouds acquired using sensors, such as points scattering due to sensors noise, or missing parts by occlusion. In addition, depending on the sensor, there are cases in which the resolution is low, and the shape can not be entirely represented. To solve these problems, the alignment of several point clouds has been attracting attention, this process is called registration. When two point clouds are acquired by changing the viewpoint, it is said that the object or the sensor have translation or rotation, that is, the point clouds are put into different locations due to the transformation parameters. This is because each point cloud is belonging to each sensor coordinate system. Registration is a method that estimates the transformation parameters between point clouds, it aligns another point cloud by transforming its coordinates system. It is noted that in this paper, we deal with point clouds without using an external sensor or texture information.

Figure 1: KPP extraction[3]

Evolutionary Computation Registration (ECR) is a method that aligns point clouds with high precision, without relying on their initial position[1]. However, this method is computationally expensive, hence, a reduction plan for the calculations is necessary[2]. Therefore, we considered the approach that extracts the characteristic parts of a point cloud to reduce the fitness computations. KeyPoint Patch (KPP) is the extraction of a sub point cloud which only selects points around a keypoint[3]. The previous approach detects an appropriate number of keypoints, then it extracts KPPs from them, and finally processes ECR between KPPs and the target point cloud.

Fig. 1 is an example of KPP extraction. The black dots are the raw point cloud and the red dots are the points that are part of the KPP. The method that uses Self-Adaptive Differential Evolution (SADE)[4] and KPP preserves the efficiency of ECR, also the processing time was greatly improved[3]. On the other hand, one dataset was converged to a local solution which has a slight registration error in all trials. Moreover, the correlation between the processing time and the registration accuracy were not evaluated.

In this paper, we proposed an improved KPP approach



(a) Truth ($RMSE = 0.0$mr). (b) Failure ($RMSE \simeq 2.0$mr).

Figure 2: Examples of registration [3]

that escapes from local solutions and accelerates the registration. First, because we considered that the local solution issue was influenced by KPPs near the occlusion of the point cloud, the proposed method detects KPPs far from the boundaries. Then, to further accelerate the process, we contemplated to reduce the number of KPPs, also to reduce the points that forms the KPP . Finally, we evaluated the performance of the proposed methods by numerical experiments.

2. PROPOSED METHOD

2.1. RELATED WORKS

In ref. [3], the conditions of the local solution issue on KPP without ECR can be reproduced as follows: the keypoint detector is Intrinsic Shape Signatures (ISS)[5], both the support radius (r_{iss}) and NMS radius (r_{nms}) are 8.0[mr], threshold Th_{21} and Th_{32} are 0.975, KPP extraction radius (r_{kpp}) is 9.0[mm] ($\simeq$ 4.5[mr]), where "[mr]" means Mesh Resolution. As for the dataset, the target point cloud is "dragonStandRight_0" and the source point cloud is "dragonStandRight_336" from the stanford 3D scanning repository[1][6], both point clouds were resampled to 2.0[mr]. Since r_{nms} is a fixed parameter, the number of KPP is numerous, that is, 20 KPPs were detected using the source point cloud.

The conditions reached by using the previous parameters are as shown in fig. 2. Fig. 2(a) is the correctly transformed point cloud with the ground truth parameters, on the other hand, fig. 2(b) is using ECR with KPP. In these conditions, the registration converged to a local solution as shown in fig. 2(b). The red dots are the target point cloud, the green dots are the source point cloud, the big green dots are the keypoints, and the big yellow-green dots are the points belonging to the KPPs. Also, the black lines that come out from the KPP show the distance to the nearest point in the target point cloud, and the length of these lines shows how big is the error. The KPP pointed by fig. 2(c) has the longest black lines, that is, we considered this patch affected the convergence of a local solution, hence must be removed.

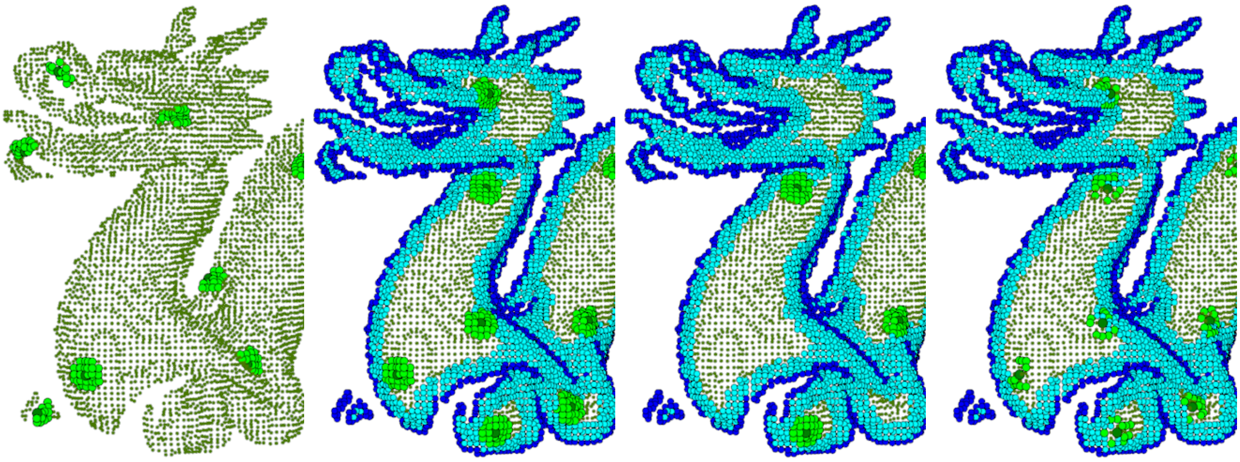
Regarding the processing time for all datasets, the number of points was reduced to an average of 14% from their original, and the processing time was reduced in about 11%. However as mentioned above, the number of KPPs was numerous, depending on datasets from 5 to 20 patches, and its relationship with the registration accuracy was not evaluated. Similarly, the number of points that are part of a KPP was fixed, and not evaluated. We considered that the processing time can be further shortened by this reduction.

2.2. BOUNDARY REMOVAL

As mentioned above, the KPP shown in fig. 2(c) must not be extracted in order to compute registration with high accuracy. In fig. 2(a), there are no red dots to match within the vicinity of fig. 2(c). Therefore, this KPP exists around a missing surface of the target point cloud. The missing surface is called occlusion; it is the surface of the object that was not exposed to the sensor (e.g., a back surface, a shadow of protruding parts). In the conventional ECR without KPP, the median value was used as the fitness computation, hence, the negative influence of the occlusion was almost ignored[7]. However, ECR with KPP used the mean to avoid falling in another local solution (bigger registration error) caused by a fewer number of KPPs. In other words, KPP is strongly affected by the occlusion.

Therefore, we assume that occlusion occurs in the neighborhood of boundaries and proposed the KPP-BR (Boundary Removal), which avoids detecting KPP near the boundaries. First, we extract the borders from the original point cloud, and remove points within the boundary isolation ra-

[1] http://graphics.stanford.edu/data/3Dscanrep/



(a) Conventional KPP. (b) KPP-BR. (c) Reduction of KPP. (d) Resampling of KPP.

Figure 3: KPP extraction and improvements.

dius (r_{iso}). Then, keypoint detection processes the remaining points. Finally, KPP-BR uses the points around the main body of the original point cloud, and far from its boundaries.

Fig. 3(b) is an example of KPP-BR, blue dots are the boundaries, and light blue dots are points within r_{iso}. Compared to the original KPP in fig. 3(a), the detected keypoints are far from the boundaries. In addition, the undesired KPP of fig. 2(c) was not detected.

2.3. KPP REDUCTION

In ref. [3], the number of selected points of a KPP was fixed. Therefore, we propose a speedup method that adjusts the number of keypoints detected by applying Non Maxima Suppression (NMS), hence reducing the number of KPP.

The KPP shown in fig. 3(c) is an example of reducing the KPP in fig. 3(b). We can confirm that the extracted KPPs from the face or the abdomen are not selected.

2.4. KPP RESAMPLING

In Ref. [3], all the points in the neighborhood of a KPP were selected. Therefore, we propose a speedup method that reduces the points in the KPP by performing resampling on it.

Fig. 3(d) is an example of random sampling by 50% the KPP of fig. 3(b). We can confirm that the number of points (in yellow-green) of the KPP are reduced.

3. EVALUATION EXPERIMENTS

3.1. DATASETS AND METRICS

In order to evaluate the proposed method, we used for the experiments the Armadillo($\mathbb{A}$), Bunny($\mathbb{B}$) and Dragon($\mathbb{D}$) models from the Stanford Repository. The target point cloud is sensed from the front, the 2 source point clouds are sensed from left to right at different positions. Furthermore, voxel resampling of 2.0[mm] was applied in advance to all the point clouds to reduce the processing time. For that purpose, we made use of the voxel resampling implementation available in the Point Cloud Library (PCL)[8]. Voxel resampling of PCL calculates the centroid of the points inside a voxel which adds some low-pass effect. The details of the datasets after voxel resampling are shown in tab. 1.

In this experiment, the number of successful registrations, registration accuracy at success and processing time are used for the evaluation. The accuracy measurement of registration is the same as in Ref. [1]; the Root Mean Squared Error ($RMSE$) of the transformed point cloud by the estimated value or truth value. Registration is regarded as a success when the $RMSE$ is smaller than the mesh resolution of the source point cloud. In other words, registration is successful when it has $RMSE < 1.0$[mr], then $RMSE$ is used as the registration accuracy. Finally, processing time is the mean value of all trials, regardless of success or failure.

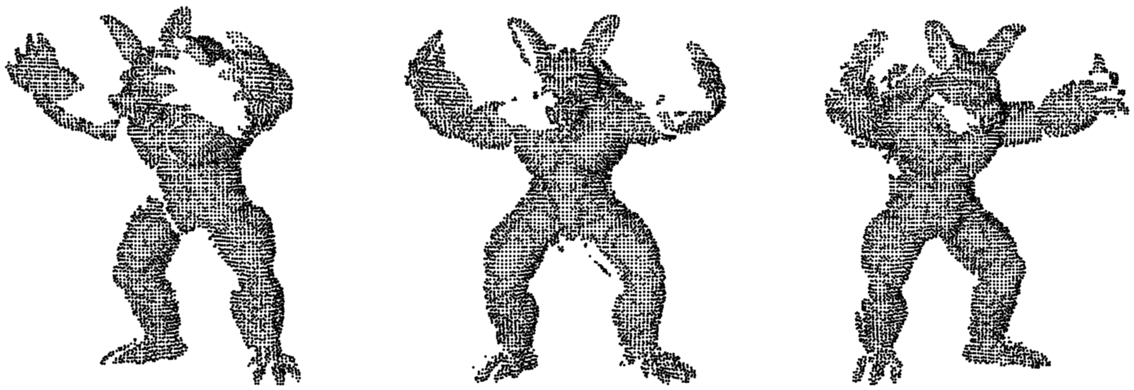
3.2. METHODS AND PARAMETERS

In this paper, we used these methods: the boundary extration is the method implemented with PCL[9], the Intrinsic Shape Signatures (ISS)[5] as the keypoint detector, and SADE[4] as the ECR algorithm. These methods, except the boundary extraction are the same as in Ref. [3], however, their parameters differ, and are as shown in tab. 2. The items that have " $*$ " on them are parameters implemented in PCL, and "Pre. ex." shows values acquired by preliminary experiments, and the details are as follows.

Although the support radius of the Boundary Extraction (r_{be}) was fixed for the entire dataset, it was able to extract boundaries with high accuracy. We increased r_{iss}

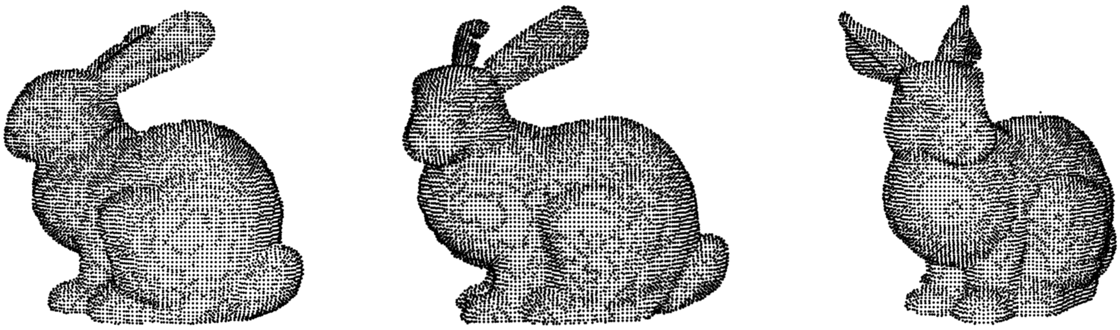
Table 1: Dataset details.

ID	Attribute	File name	Voxel [mm]	mr [mm]	Points	Angle	View
$\mathbb{A}_1$	Target	ArmadilloStand_180	2.0	1.3	5292	-	Fig. 4(b)
$\mathbb{A}_2$	Source	ArmadilloStand_210	2.0	1.3	5095	$+30$	Fig. 4(a)
$\mathbb{A}_3$	Source	ArmadilloStand_150	2.0	1.4	5559	-30	Fig. 4(c)
$\mathbb{B}_1$	Target	bun000	2.0	1.4	7133	-	Fig. 5(b)
$\mathbb{B}_2$	Source	bun045	2.0	1.4	6813	$+45$	Fig. 5(a)
$\mathbb{B}_3$	Source	bun315	2.0	1.3	6831	-45	Fig. 5(c)
$\mathbb{D}_1$	Target	dragonStandRight_0	2.0	1.3	7155	-	Fig. 6(b)
$\mathbb{D}_2$	Source	dragonStandRight_24	2.0	1.3	6312	$+24$	Fig. 6(a)
$\mathbb{D}_3$	Source	dragonStandRight_336	2.0	1.3	7512	-24	Fig. 6(c)



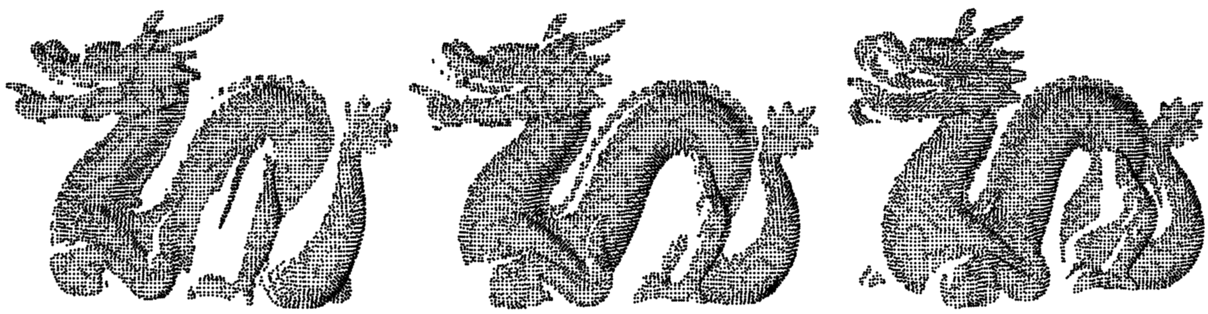
(a) $\mathbb{A}_2$ (ArmadilloStand_210). (b) $\mathbb{A}_1$ (ArmadilloStand_180). (c) $\mathbb{A}_3$ (ArmadilloStand_150).

Figure 4: Views of the armadillo.



(a) $\mathbb{B}_2$ (bun045). (b) $\mathbb{B}_1$ (bun000). (c) $\mathbb{B}_3$ (bun315).

Figure 5: Views of the Bunny.



(a) $\mathbb{D}_2$ (dragonStandRight_24). (b) $\mathbb{D}_1$ (dragonStandRight_0). (c) $\mathbb{D}_3$ (dragonStandRight_336).

Figure 6: Views of the dragon.

Table 2: Parameters

Method	Parameter	Value	Unit	Remark
Bound. ext.	$r_{be}{}^*$	4.0	mr	Pre. ex.
	Angle thresh.*	90.0	deg	Ref. [9]
ISS	$r_{iss}{}^*$	10.0	mr	Ref. [10]
	$Th_{21}{}^*$	0.600	-	Pre. ex.
	$Th_{32}{}^*$	0.975	-	Ref. [5]
	$r_{nms}{}^*$	-	mr	Ajusted
KPP	r_{kpp}	4.0	mr	Pre. ex.
	r_{iso}	5.0	mr	Pre. ex.
ECR	Population	30	-	Ref. [3]
	Rotation	± 180	deg	Ref. [1, 3]
	Translation	± 0.04	m	Ref. [1, 3]
Termination	Diff. fitness	1.0e-10	m^2	Pre. ex.

Table 3: Specification of the computer

CPU	Intel Core i5 4570S (3.50GHz)
RAM	8.0 [GBytes]
OS	Ubuntu version 14.04.3 (64bit)
Compiler	Clang++ version 3.4
Library	PCL version 1.7.2

larger than ref. [3] in order to increase the balance of the repeatability[10]. In Ref. [5], Th_{21} of ISS is encouraged to be set at 0.975. However, with this value it detects numerous keypoints on planar surfaces. For that reason, we decided to detect keypoints only in protruded locations by being more strict on the thresholds. Moreover, by adjusting r_{nms}, we obtained 10 keypoints (i.e., the number of KPP is 10 patches). The extraction r_{kpp} is set to a similar value of [3] and equals to r_{be}. Moreover, we defined r_{iso} as a value larger than the KPP extraction. Also, we implemented ECR with SADE based on PCL, and the control parameters of SADE are the same as Ref. [4]. The finishing condition was replaced by the difference of the fitness value between generations, instead of the number of generations as in [3] to prevent registration to stop during convergence. When the value between the current and previous fitness is smaller than this parameter, ECR is terminated. Notice that we used the squared meter as the unit to avoid increasing the computational costs of the fitness function. We think that sufficient convergence can be obtained when this parameter is smaller than 0.01[mr].

Finally, we processed 10 times each experiment with the computer shown in tab. 3

3.3. KPP-BR PERFORMANCE EVALUATION

In this experiment, we validate the performance of KPP-BR. Tab. 4 shows the results of ECR using KPP, and KPP-BR. In the first row, "Dataset" is the combination of point clouds, "Method" is the approach of KPP extraction, "Proc. points" is the mean number of points constructing the KPPs,

Table 4: Result of KPP-BR performance.

Dataset	Method	Proc. points	Success No.	Success $RMSE$	Time [sec]
$\{\mathbb{A}_1, \mathbb{A}_2\}$	KPP	234	10/10	0.321	7.098
	KPP-BR	268	10/10	0.209	9.357
$\{\mathbb{A}_1, \mathbb{A}_3\}$	KPP	252	10/10	0.904	7.305
	KPP-BR	287	9/10	0.504	7.989
$\{\mathbb{B}_1, \mathbb{B}_2\}$	KPP	267	10/10	0.355	8.814
	KPP-BR	302	9/10	0.274	9.764
$\{\mathbb{B}_1, \mathbb{B}_3\}$	KPP	237	0/10	(4.706)	5.887
	KPP-BR	284	10/10	0.653	12.111
$\{\mathbb{D}_1, \mathbb{D}_2\}$	KPP	201	10/10	0.271	6.276
	KPP-BR	269	9/10	0.113	11.036
$\{\mathbb{D}_1, \mathbb{D}_3\}$	KPP	198	0/10	(7.241)	6.920
	KPP-BR	284	10/10	0.238	10.009

"Success No." is the number of the successful registrations, "Success $RMSE$" is the mean value of the successful $RMSE$, and "Time" is the mean computational time. When the whole registration fails, the median value of the trials is described with parentheses in the "Success $RMSE$" column.

The clear differences between using the current parameters and the ones described in ref. [3], which uses conventional KPP, are the following (underlined values): $\{\mathbb{B}_1, \mathbb{B}_3\}$ totally fell into local solutions, and the $RMSE$ of $\{\mathbb{D}_1, \mathbb{D}_3\}$ was increased in about 4 times. This is because the impact of occlusion is increased by reducing the number of KPPs. On the other hand, we confirmed that using KPP-BR, the success ratio is high in the entire dataset and its $RMSE$ is superior than the conventional KPP. Therefore, the proposed method was able to extract a smaller number of KPPs in more suitable positions.

Regarding the processing time, KPP-BR increased in about 1.4 times when compared to the conventional KPP. This increment is presumably because KPP-BR is not detecting keypoints in hidden parts by occlusion. Therefore, we can say that the number of processed points increased.

From the above, we confirmed that KPP-BR slightly increased the processing time, but also confirmed it has high success ratio and high registration accuracy.

3.4. PERFORMANCE IMPROVEMENT

In this experiment, we measured the number of KPPs and observed that their structure was reduced. Furthermore, we used the same parameters of the previous experiment. Regarding the reduction in the number of KPPs, we manually adjusted the r_{nms} to extract a fixed number of KPPs; $\{10, 8, 6, 4\}$. In order to reduce the structure of 10 KPPs we applied random sampling of $\{100, 75, 50, 25\}[\%]$. The results of reducing the number of KPP are shown in tab. 5(a), and the results of reducing the structure of the KPP are

Table 5: Performance improved in KPP-BR.

(a) Effect of KPP reduction.

Dataset	No. KPP	Proc. points	Success No.	Success $RMSE$	Time [sec]
$\{\mathbb{A}_1, \mathbb{A}_2\}$	10	268	10/10	0.209	9.357
	8	211	9/10	0.256	7.123
	6	156	9/10	0.307	5.106
	4	106	<u>3/10</u>	0.223	4.104
$\{\mathbb{A}_1, \mathbb{A}_3\}$	10	287	9/10	0.504	7.989
	8	228	9/10	0.467	7.443
	6	173	10/10	0.578	6.419
	4	114	<u>6/10</u>	0.630	4.265
$\{\mathbb{B}_1, \mathbb{B}_2\}$	10	302	9/10	0.274	9.764
	8	242	9/10	0.388	7.812
	6	180	10/10	0.443	6.437
	4	120	<u>4/10</u>	0.549	5.985
$\{\mathbb{B}_1, \mathbb{B}_3\}$	10	284	10/10	0.653	12.111
	8	233	10/10	0.597	9.455
	6	172	10/10	0.697	7.349
	4	115	<u>2/10</u>	0.778	6.508
$\{\mathbb{D}_1, \mathbb{D}_2\}$	10	269	9/10	0.113	11.036
	8	210	9/10	0.107	7.375
	6	158	8/10	0.127	6.020
	4	101	<u>6/10</u>	0.437	4.478
$\{\mathbb{D}_1, \mathbb{D}_3\}$	10	284	10/10	0.238	10.009
	8	227	9/10	0.291	7.641
	6	170	8/10	0.297	6.139
	4	115	<u>2/10</u>	0.677	4.645

(b) Effect of KPP resampling.

Dataset	Ratio [%]	Proc. points	Success No.	Success $RMSE$	Time [sec]
$\{\mathbb{A}_1, \mathbb{A}_2\}$	100	268	10/10	0.209	9.357
	75	210	10/10	0.237	6.646
	50	147	10/10	0.282	5.359
	25	70	10/10	0.374	2.720
$\{\mathbb{A}_1, \mathbb{A}_3\}$	100	287	9/10	0.504	7.989
	75	224	9/10	0.477	7.544
	50	155	8/10	0.428	5.125
	25	74	7/10	0.462	2.437
$\{\mathbb{B}_1, \mathbb{B}_2\}$	100	302	9/10	0.274	9.764
	75	236	10/10	0.319	7.585
	50	163	10/10	0.397	6.025
	25	78	10/10	0.416	2.640
$\{\mathbb{B}_1, \mathbb{B}_3\}$	100	284	10/10	0.653	12.111
	75	221	10/10	0.637	9.180
	50	154	10/10	0.624	6.761
	25	74	10/10	0.631	3.073
$\{\mathbb{D}_1, \mathbb{D}_2\}$	100	269	9/10	0.113	11.036
	75	210	9/10	0.121	8.091
	50	147	9/10	0.165	5.421
	25	70	9/10	0.291	2.652
$\{\mathbb{D}_1, \mathbb{D}_3\}$	100	284	10/10	0.238	10.009
	75	222	10/10	0.214	7.800
	50	154	10/10	0.277	5.704
	25	73	9/10	0.286	2.674

shown in tab. 5(b).

By reducing the number of KPPs results in an acceleration of the processsing time, however, as the number of KPPs is reduced the number of success drops. Particularly, the results of 4 KPPs were terrible (underlined values). Therefore, we can say that the processing time of this method is a trade-off with the success ratio. On the other hand, by reducing the number of points of a KPP, we can gain speed while almost keeping the accuracy and the number of successes. The resampling ratio was directly proportional to the speed gain ratio.

From the above, we confirmed that both KPP reduction, and KPP resampling result in speed gain. However, we recommend using KPP resampling, since the KPP reduction clearly decreased the success ratio. Particularly, even when reducing the number of points in a KPP by 25%, the success ratio and the registration accuracy are not significantly affected, hence we can say that this is an effective method.

4. CONCLUSION

In this paper, we proposed the following improvements to KPP in order to speedup ECR. First, to reduce the negative influence of occlusion, KPP-BR is extracted farther from boundaries. Then, to accelerate the process, the number of KPPs and its points were reduced.

From the results of the experiments, we confirmed that KPP-BR can succeed in the same datasets in which the conventional KPP failed[3]. KPP reduction was able accelerate ECR, however, we hardly estimate the appropriate number of KPP in advance. On the other hand, KPP resampling accelerated processing time while maintaining registration accuracy. The combination of KPP-BR and KPP resampling was able to reduce the negative influence of occlusion, moreover it was 2.6 times faster than the conventional KPP. Therefore we can say that the proposed method results in a more effective ECR.

[1] J. Santamaría, O. Cordón, and S. Damas: “A comparative study of state-of-the-art evolutionary image registration methods for 3d modeling,” Computer Vision and Image Understanding, vol.115, no.9, pp.1340–1354 (2011).

[2] J. Salvi, C. Matabosch, D. Fofi, and J. Forest: “A review of recent range image registration methods with accuracy evaluation,” Image and Vision computing, vol.25, no.5, pp.578–596 (2007).

[3] K. Uenishi, M. Iwakiri, and K. Tanaka: “Efficient Point Clouds Registration Based on Differential Evolution with Keypoint Patches,” proc. International Workshop on Image Electronics and Visual Computing (2014).

[4] J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer: “Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems,” IEEE Trans. Evolutionary Computation, vol.10, no.6, pp.646–657 (2006).

[5] Y. Zhong: “Intrinsic shape signatures: A shape descriptor for 3D object recognition,” Proceedings of International Conference on Computer Vision Workshops, pp.689–696 (2009).

[6] G. Turk and M. Levoy: “Zippered polygon meshes from range images,” Proceedings of the 21st annual conference on Computer graphics and interactive techniques, pp.311–318, ACM (1994).

[7] M. Rodrigues, R. Fisher, and Y. Liu: “Special issue on registration and fusion of range images,” Computer Vision and Image Understanding, vol.87, no.1, pp.1–7 (2002).

[8] R.B. Rusu and S. Cousins: “3D is here: Point Cloud Library (PCL),” Proceedings of IEEE International Conference on Robotics and Automation, pp.1–4, IEEE (2011).

[9] R.B. Rusu:, Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments, Ph.D. thesis, Computer Science department, Technische Universitaet Muenchen, Germany (2009).

[10] F. Tombari, S. Salti, and L. Di Stefano: “Performance Evaluation of 3D Keypoint Detectors,” International Journal of Computer Vision, vol.102, no.1-3, pp.198–220 (2013).