KPPF:
Keypoint-based Point-Pair-Feature for scalable automatic global registration of large RGB-D scans

L. Malleus[1], T. Fisichella[1], D. Lingrand[1], F. Precioso[1], N. Gros[2], Y. Noutary[2], L. Robert[2], L. Samoun[1]
[1] Université Côte d'Azur, CNRS, I3S, France [2] Autodesk, Sophia Antipolis, France
{malleus,lingrand,precioso}@i3s.unice.fr

Abstract

One of the most important challenges in the field of 3D data processing is to be able to reconstruct a complete 3D scene with a high accuracy from several captures. Usually this process is achieved through two main phases: a coarse, or rough, alignment step then a fine alignment. In this article we propose an automatic scalable global registration method (i.e. without arbitrary pose of the sensor) under the following constraints: markerless, very large scale data (several, potentially many millions of points per scans), little overlap between scans, for more than two or three dozens of scans, without a priori knowledge on the 6 degrees of freedom.

Here we only address the coarse alignment, and consider the fine alignment step tackled by dedicated existing approaches such as Iterative Closest Point (ICP) [3].

We evaluate thoroughly our method on our own dataset of 33 real large scale scans of an indoor building. The data presents some pairs of scans with very little overlap, architectural challenges (a patio and a rotunda open through several levels of the buildings, etc), several millions of points per scan. We will make this dataset public as part of a benchmark available for the community.

We have thus evaluated the accuracy of our method, the scalability to the initial amount of points and the robustness to occlusions, little scan overlap and architectural challenges.

1. Introduction

Nowadays, the world of 3D data is facing a true revolution with many new devices easily available for both capturing and visualizing data whether it be pure virtual 3D data for virtual reality applications or combined 3D and visual data for augmented reality. This new context with everyday new applications based on 3D information, 3D data of many different types, new sensors, comes up with also a lot of new challenges. Among these challenges, the first processing step preceding almost always any other processing from 3D data is the registration of data acquired by a sensor from different locations. For professional quality even more for large scale environments, LiDAR technology remains the first choice. LiDAR acquisitions provide us with point clouds with little noise and the knowledge of the spatial range. Indeed, whatever the kind of sensors considered (Laser scanner, Kinect...), no existing technology can acquire the whole 3D information of a scene all at once.

A 3D acquisition usually results in a set of 3D points (a 3D point cloud), with or without associated RGB information depending on the sensor, covering partially the spatial area of the full scene. Single scans, from different sensor positions, are acquired within a local coordinate frame defined by the instrument. For visualization and further data processing of the point cloud the single scans must be transformed into a common coordinate frame. This process is usually called as registration. Given two or more 3D point clouds that have a subset of points in common, the goal of 3D registration is to compute the rigid transformation that aligns these point clouds, providing an estimate of the relative pose between them. This registration of partially overlapping 3D point clouds, usually considered pair by pair, taken from different views is thus an essential task in 3D computer vision and the basis of many algorithms in 3D vision applications. There are two well identified challenges in this task of 3D registration:

- Global registration which refers to the registration of a pair of 3D point clouds without initial guess on the pose of one point cloud to the other, the pose is thus arbitrary.
- Local registration which refers to the registration of a pair of 3D point clouds from a valid initial estimate of the pose of between the two clouds.

Many algorithms for registration have addressed these challenges in the last decades. It is commonly accepted in

2473-9944/17 $31.00 © 2017 IEEE
DOI 10.1109/ICCVW.2017.294

the community to classify the algorithms into 2 distinctive classes [6] regarding to which challenge they address:

- *Coarse registration*: it consists in roughly aligning two scans without any clue about the initial alignment.
- *Fine registration*: from a given coarse alignment, it refines the result, generally by minimizing error iteratively.

The global registration usually requires both coarse and fine registration steps, while for the local registration challenge it is often sufficient to consider a fine registration process only.

A popular type of approach involves iterative algorithm, Iterative Closest Point (ICP) [3] and variants [14, 9, 12]. In practice, the original ICP algorithms tend to converge poorly when subjected to severe noise and large pose displacements without a good initial guess on scans alignment. This algorithm, or its variants, is then often used for a fine registration stage in order to improve previous coarse registration but is not suitable as a direct solution for coarse registration.

Some other approaches try to get rid of the requirements of good initial estimate by extracting invariant local features and finding correspondences. The main limitation of such methods is a lack of robustness to large baseline (large sensor viewpoint changes) and little overlap.

With the availability of several new consumer-grade 3D sensors, many solutions have been proposed to solve this problem and work pretty well for reasonably sized point clouds (few thousands of points) even with little overlap, noise and pretty large baseline.

LiDAR scans have several advantages compared to other point clouds:

- They generally have very little noise, compared to other point clouds capture with sensors like Kinect;
- They are already at scale contrary to point clouds produced by structure from motion. This reduces the complexity of the problem by locking the scale.

Many algorithms try to solve this problem using either only geometrical information, or only visual information (e.g. RGB values attached to the 3D points), and eventually some use both. Most of the current solutions are not able to solve efficiently the global registration problem when all the following constraints are present: large scale data, large baseline, little overlap, no initial guess of the relative alignment.

Here, we present a keypoint-based approach. This means we look for particular points/regions scan that are likely to be repeatable across scans. Then we retrieve information from those points and their close neighborhood (position and normal) and build a descriptor on it. One correspondence is sufficient to estimate a rigid transformation between the two scan locations. Our goal is to solve the 3D global registration problem in environments with large scale data (several millions of points), large baseline, little overlap and no prior knowledge on the alignment between scans.

2. Related work

A survey of the most significant contributions in the domain can be found in Diez *et al*. [6]. As described in this survey, all the approaches of coarse registration can be compared considering the main steps:

- Salient information detectors that aims at detecting keypoints to reduce the size of the representation of the data such as MeshDoG [17], or Harris3D [15]
- Salient information descriptors in order to extract relevant and discriminant information from the previously detected keypoints such as Improved Spin Image [19] or Rotational Projection Statistics [8].
- Refinement: This step is mainly based on ICP [3] and its improvements [14, 9, 12], which improves the coarse alignment for a better fit.

This survey is not only reporting the current state-of-the-art on coarse registration but it proposes a pipeline to classify these existing methods with respect to the three aforementioned main steps. This survey also defines a standard formal notation to provide a global point of view of the state-of-the-art.

The reader looking for more details on the existing methods on coarse registration should consider this survey.

Our method is fully in line with the previous three steps: our idea is to detect keypoints from the point clouds/meshes, to describe them exploiting the (X,Y,Z) information (no RGB) and to create correspondences which are finally used to generate hypotheses of possible registrations.

In that context, one classical method to describe the scans around salient detectors is based on normal distribution. This idea has already been taken into account in several variants. For instance, an algorithm based on normals distribution has been proposed by Makadia *et al*. [9]. Normals are accumulated in an orientation histogram. Peaks are identified and create constellations, that are then aligned to directly give the rotation between two scans. Translation can be deduced in a further step. The limitations identified in this approach are that overlap has to be important, and that normals from the overlap area must be discriminative regarding the distribution of normals through the whole point clouds, while being consistent with this distribution.
One solution to address these limitations, when RGB information is available, is to combine 3D and RGB detection,

description and matching. For instance, Zeisl *et al*. [18] combine 3D information with standard vision matching using 2D SIFT. The idea in their paper is to project the scan in perspective views orientated towards "salient directions", which are identified as peaks on the normals distribution as in the previous work. However, this algorithm is challenged by point clouds with few planar areas, or surfaces presenting an important tilt when viewed from the scan location.
In cases such as scanner acquisitions, there may be lot of self-occlusions, leaving large parts of the scene hidden from the scanner. Since the occlusions are likely to be different from a viewpoint to another, hence from one scan to another, using a larger size for the neighborhood might be not suitable, as the information held by this neighborhood could vary a lot.

This led us to build a descriptor purely based on 3D information and that would not make strong assumption on the point neighborhood. Moreover the detector of salient points used in our algorithm uses RGB or intensity information in the neighborhood, since it is known to be quite resilient to occlusions and differences in sampling density as stated in Theiler *et al*. [16].

The seminal work to our approach is the work from Aiger *et al*. [1]. They proposed the 4-Points Congruent Sets (4PCS) algorithm that does not use local description. Instead, it considers groups of 4 quasi-coplanar points in the point cloud. It uses a shape descriptor based on the distances between these points and the intersection of the line segments they define.

The selected points must be spread enough to guarantee algorithm stability, and close enough to ensure that the points are in the overlap area between the two viewpoints. The geometric configuration of four coplanar points is discriminant enough to create strong matches between the two point clouds.
This work on 4PCS algorithm has inspired several other recent approaches still introducing the 4-Points Congruent Sets principle: the very recent Super4PCS from Mellado *et al*. [10] is based on a smart indexing data organization reducing the algorithm complexity from quadratic to linear in the number or data points; Mohamad *et al*. [11] generalizes 4PCS by allowing the two pairs to fall on two different planes which have an arbitrary distance between them, then by increasing this distance, the search space of matching hypotheses exponentially leading to a run-time improvement of up to 80%.

However the Super4PCS is more complex to implement and the runtime improvement is generally of 50% compared to 4PCS. Similarly the Generalized 4PCS reach more than 80% of runtime improvement but this is only evaluated on datasets with less than 100K points.

Theiler *et al*. [16] and more recently Bueno *et al*. [5] cut down the complexity by using keypoint detection as a previous step to the 4PCS algorithm. This allows to work with larger points clouds.

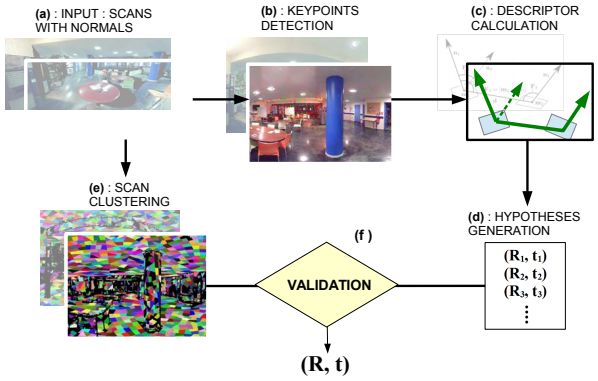

Figure 1. Overview of the algorithm. Each step will be explained further and referenced with the associated letter

Another idea to reduce the complexity of 4PCS is to rely on pairs of points rather than on 4-Points Congruent Sets.

The idea of a four-component descriptor based on angles and distance between two points, has been first proposed by Drost *et al*. [7]. They applied this descriptor, called Point Pair Feature (PPF), for efficient and robust 3D object recognition.

Then, Birdal and Ilic [4] proposed a full processing chain based on their PPF descriptor to address the problem of large scene scan registration. Even though their approach may seem similar to ours, our approach differs from Birdal's method in several significant aspects that we are going to describe in the following section. It mainly consists in the introduction of keypoints detection to deal with large point clouds, a box filtering associated, a different voting space based on translations, and the hypotheses verification step based on hierarchical clustering.

3. Algorithm overview

An overview of the algorithm is presented in figure (1).

Our algorithm proceeds by first extracting keypoints locally in each scan, grouping them in pairs and building a descriptor for each pair.

Then, the descriptors are used to estimate rigid transformations. Since there may be a lot of outliers, we accumulate those transformations into a voting space and look for zones with high density - the idea being similar to a classic Hough transform. The most relevant transforms are then scored to define the correct alignment.

3.1. Detector: SIFT3D (figure 1 b)

Numerous 3D detectors are available on the benchmark. We need a detector that is not too sensitive to self-occlusions and has a good repeatability, regardless of the descriptor. Moreover, since we will use normals in the descriptor, the detected points must have a well-defined normal, which means features such as corners are not good candidates.

For this reason, we decided to extract keypoints based on RGB or reflectance data attached to the scan, rather than on geometry itself.
We did tests on the repeatability of common detectors on our datasets: SIFT3D, Harris3D and Intrinsic Shape Signature. SIFT 3D showed to be the most suitable detector for our purpose. We used the implementation provided by Point Cloud Library (PCL). To counter the effects of occlusions on the detection step, the maximum scale is set to a low value compared to the size of the point cloud. To ensure that keypoints are detected in every part of the scan, we use a low initial threshold on response intensity (in practice, we use a minimum contrast of 0.1). Then, to control the total number of keypoints, we split the space in fixed-size 3D buckets and only consider the first n strongest keypoints in each bucket. In practice, we use buckets sizing 2 x 2 x 2 meters with 4 keypoints in each. It ensures there are not too few keypoints, which would lead to a lack of repeatability, and not too many, which would drastically increase the execution time.

3.2. The descriptor: KPPF (figure 1 c)

Our algorithm is inspired by the K-4PCS and the PPF. The driving idea behind it is that 4PCS has issues when dealing with large point clouds: complexity can make it intractable to use all input data points. By limiting computation to keypoints detected by a feature detector such as SIFT, the point clouds are down-sampled in a way that preserves correspondences if features are repeatable enough. And since, finding sets of four coplanar keypoints that would be visible from both point clouds can be challenging due to the non-repeatable features detected, we reduce the number of points in a descriptor to two. To still be able to compute the rigid transformation from one pair of descriptors, we keep the normal associated to both points.

3.3. Descriptor construction

Keypoint detection: SIFT 3D undertakes successive filters at different scales, that are then subtracted. Maxima are identified as keypoints. To limit sensitivity to occlusions, we voluntarily forced the maximum scale not to be too large regarding the point cloud size.

Normals extraction: At every keypoint, we need a reliable normal information. For this, we consider a neighborhood around this keypoint and fit a plane to the corresponding data using normalized least squares [2]. If the neighborhood is planar enough, a robust normal is calculated and refined to fit at best the local geometry. Keypoints that do not have such normals are discarded and ignored in the subsequent stages (see red points in figure 2).

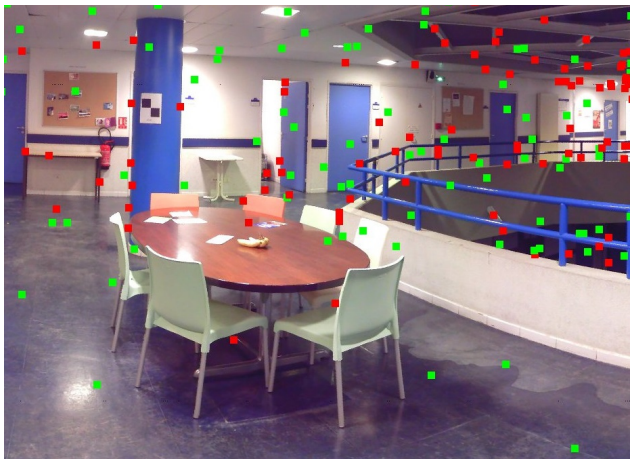

Figure 2. Close-up of the keypoints extracted in scan number 12. Green dots have stable normals whereas red do not - and are thus discarded.

Descriptors calculation: The descriptor consists of values derived from a pair of two keypoints and their normals, like in the PPF descriptor [7]. At this stage, we consider all the points that have been detected at the first step using 3D SIFT, and have not been discarded at the normal estimation step. Then, we estimate the 4 following values that build the descriptor:

- spatial distance between the two points (1 value)
- angles between the normals and the line defined by the two points (2 values) [1]
- angle between the normals themselves (1 value)

Such a descriptor is fast to compute and very compact. This representation is very convenient and conveys sufficient information to compute a rigid transformation from a pair of descriptors.

[1] It is there important to use a criterion to sort the two angles that would be shared through different viewpoints. If we do not, every set of two points could have two valid descriptors and it would drastically enhance the complexity. We chose to set the smallest as first.

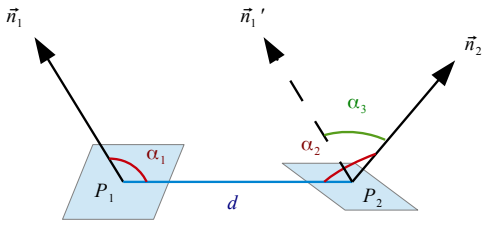

Figure 3. The descriptor itself: angles between the normals and the line defined by P_1 and P_2 (α_1 and α_2) and angle between the normals $\vec{n_1}$ and $\vec{n_2}$ (α_3).

3.4. Descriptors matching

The algorithm proceeds by considering, for all descriptors in scan 1, the potential candidates in scan 2. This is done finding the nearest neighbors in the descriptor space. To efficiently find neighbors, a 4D kd-tree is built using all the descriptors for each point cloud. For any descriptor from a given point cloud, we can efficiently look for the best candidates in the other point cloud by doing an approximate nearest neighbor search in the 4D kd-tree. The candidate pairs of descriptors are used in a voting scheme presented in the following section 3.5.

In the 4D kd-tree, we are inserting 4-vectors made of 3 angles and distance. Since the kd-tree relies on L2 distances between these 4-vectors, we have to normalize the data so that an increment on any of the four values would change the distance between descriptors consistently: we tested using either raw cosines of angles, raw angles, and angles with a scalar factor (to fit with the distance dimension). The best configuration is raw angles in degrees, and distance in meters multiplied by 40.

3.5. Hypothesis generation:(figure 1 d)

From all the candidate pairs considered, we derive the strongest rotation-translation hypotheses, i.e. the ones that get most support from the candidate pairs. To achieve this, we use a voting scheme based on 3D translation. A regular 3D voxel space is generated with an appropriate range and scale regarding the point cloud. The three dimensions represent respectively the x, y and z components of the translations. In practice, we use 50 x 50 x 50cm voxels. In each cell, we create and maintain a list of rotation clusters. The pseudo-code below describes how a candidate pair is processed:

To determine if a candidate belongs to a "cluster", we first compute the product of the candidate rotation matrix with the inverse of the representative of the cluster. Then we check if the angle (fourth component of the quaternion) is smaller than a threshold (in practice we use 5°).
To handle the fact that translations may land near a boundary between two voxels, each translation also votes for the 26 voxels around, with a score determined by a Gaussian of its distance to the center of the voxel.

Data: Unclassified transform (rotation + translation) candidate
Result: Transform candidate stored in the voxel space
Construction of the voxel space of translations;
while *not empty candidates pool* **do**
  read the transform of the candidate (hypothesis);
  select the voxel corresponding to the translation;
  if *voxel is empty* **then**
    store the hypothesis as a full transform;
  else
    if *hypothesis' rotation close enough to one of the rotation clusters'* **then**
      add the hypothesis to the cluster;
    else
      create a new cluster and add hypothesis;
    end
  end
end

Algorithm 1: Storing a candidate in the voxel space

3.6. Hypothesis validation:(figure 1 f)

To accept or reject a hypothesis, we produce a higher level representation of the original point cloud data of each scan using the adaptive hierarchical clustering algorithm proposed by Pauly *et al*. [13]. Once patches are built, we estimate the ratio between the number of matches and no-matches. First, a patch from a scan is picked and placed in the other scan according to the hypothesis. Rays are cast from the center to determine the corresponding patch in the scan. If it lands in front of the other patch, it is a no-match, because the other patch should be occulted. If it lands behind the other patch, we cannot say anything. If it lands close to the other patch, it is a match if normals correpond, a no-match if they do not. Ratios are evaluated in both directions (from one scan to another and vice versa), and a threshold is applied to validate the hypothesis.

4. Experiments

4.1. Data

We acquired the dataset used in this study using a FARO®Focus3D Laser Scanner. The acquisition angles range from -60°to + 90°for elevation, and -180°to +180°for azimuth. The resolution of the spherical images associated is 5142 x 2134. The dataset is made of 33 scans located in

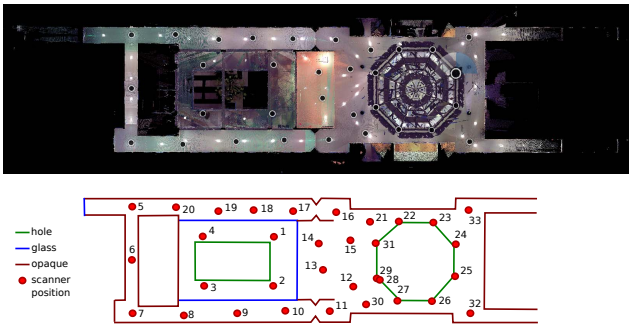

Figure 4. Top view of the scanner positions of the dataset

the positions indicated on the figure 4 (black spots on top, red on bottom). To estimate the ground-truth positions, we used a manual registration method. Ground-truth is used to assess the performance of our algorithm in the next paragraphs.

4.2. Overlap

In our assessment, we have estimated the overlap for all pairs of scans. This gives information about how likely the scans are to match.
To estimate this criterion, we represent each scan with voxels of fixed size: 20 x 20 x 20 cm. A voxel is activated if it contains at least one point. Then, the two scans are put in a common frame using the ground-truth transform. Points from one scan are projected into the voxel space of the other scan, through the pairwise ground-truth transform. The overlap is the ratio between the voxels activated by both scans and the voxels activated by only one scan. For the 528 pairs of scans, we observe an overlap mean value of 9% with high standard deviation of 11%. Values of overlap for each pair of scans are plotted on figure 5 and the distribution of overlap values on figure 6. We observe that 29% of the pairs of scans have an overlap ratio less than 2% that could be explained by occlusion in the scene (see figure 4). We also observe that 4.5% of the pairs of scans have more than 30% of overlap and only 1.3% have more than 50% of overlap.

4.3. Results

4.3.1 Repeatability of the detector

The repeatability of detected keypoints is systematically estimated using the same process: each keypoint is reprojected in the other scan using ground-truth transform. If it lies in the area of the scan (if there is a 3D point at less than 5 centimeters), we check if there is a keypoint at less than 20 centimeters. The repeatability is the ratio between the number of points that fill this last criterion and the total number of keypoints that were successfully reprojected. Repeatability has been computed for each pair of scans

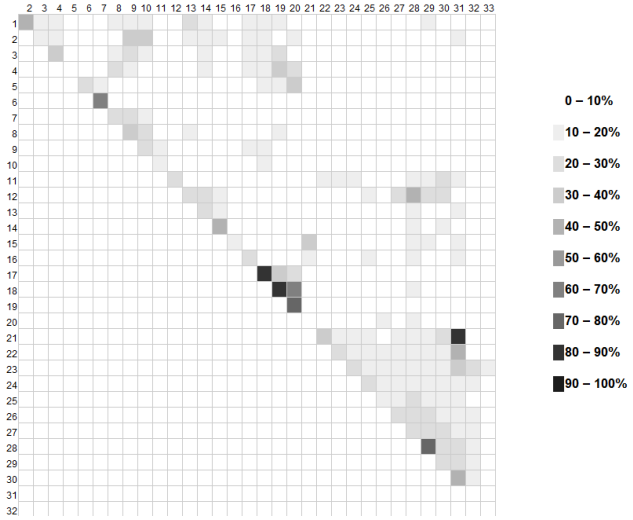

Figure 5. Overlap percentage between 2 scans. Mean overlap of 9%, standard deviation of 11%.

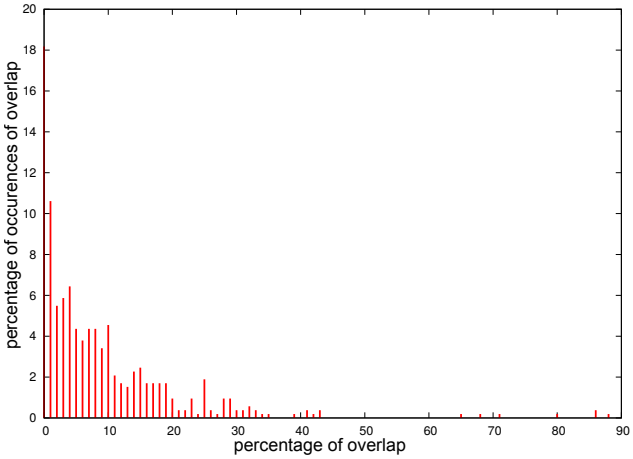

Figure 6. Distribution of the percentage of overlap between scans.

that overlap in two directions (as for example scan 2 with respect to scan 4 and scan 4 with respect to scan 2).
As a starting point, SIFT keypoints are detected, using the following values as parameters: minimum scale: 0.2, number of octaves: 4, number of scales per octave: 8 and response threshold: 0.1. This last parameter impacts the number of keypoints, thus increasing the repeatability and the complexity of the algorithm. We evaluated the repeatability of keypoints as well as the the performance of the algorithm.

For the 33 scans (see paragraph 4.1), the mean repeatability is 0.144 with a standard deviation of 0.065. This means only one out of seven keypoints from a scan will find a match in the other one, which is really low and detrimental. We will see in further experiment that even with low

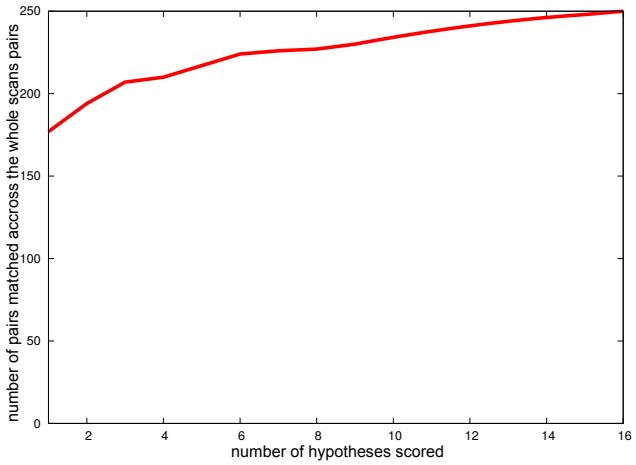

Figure 7. Number of pairs successfully matched depending on the number of top-rated hypotheses over 33*32 = 1056 pairs.

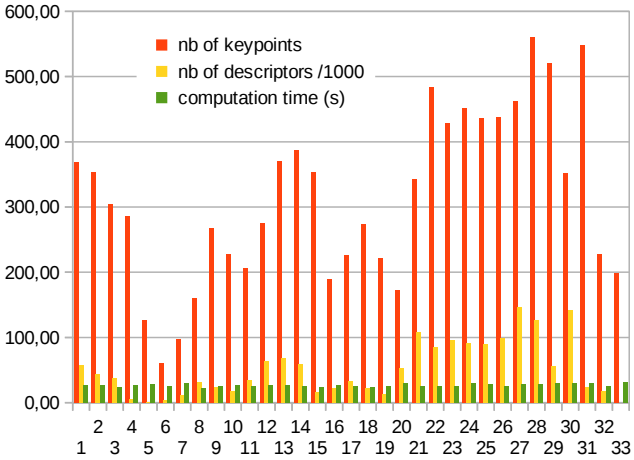

Figure 8. Number of keypoints, number of descriptors (divided by 1000 to be displayed on the same graph), time of extraction, for each of the 33 scans.

repeatability, our algorithm still has good performance.

4.3.2 Pertinence of the descriptor/voting scheme

The voting scheme gives us a list of hypotheses sorted in decreasing confidence order. The ones with the highest score are the ones that appeared the most during the voting process, and are likely to be the most relevant. We only score the first 16 ones, since considering more does not bring a lot of improvement, as illustrated on the Figure 7: it represents the number of pairs of scans that successfully matched depending on the number of hypotheses considered. If only one hypothesis is considered, the best transformation estimated between two scans is applied. If more than one hypotheses are considered, for each pair of scans, all corresponding possible transformations are tested and best results are selected. Adding more hypothesis increase thus the number of pairs successfully matched. However, as shown in figure 7, the increasing rate of matching is quite low. Considering that adding hypothesis is costly, we may limit the number of hypotheses to be tested.

4.3.3 Computation time for data preprocess and matching

Our algorithm is splitted into two parts:

- The first one extracts information - descriptors - from the scans.
- The second one matches those descriptors across scans.

Step 1: Information extraction from scans This step extracts information, keypoints and descriptors, independently from each scans. The number of informations and computation time are displayed in figure 8. The computation time appears to be independent of the number of informations with a mean of 27 seconds and a standard deviation of 2 seconds.

Step 2: Matching The time is highly dependent on the number of keypoints extracted/descriptors built as illustrated in figure 9. The computation time has a mean value of 2.17 and a standard deviation of 2.73. The average quality is of 144 with a standard deviation of 327.

Comparison to K4PCS The algorithm that was closest to ours in terms of method was K4PCS. We decided to test this algorithm on our dataset. The total number of scans pairs is 528. In order to be able to use K4PCS, we need an estimation of the overlap between each pair. For our experiments, we used the overlap values presented on Figure 5. The results are displayed in the Table 1 below.

-	K4PCS	KPPF
Number of scans pairs matched	28	250
Time per scans pair	135s	145s

Table 1. Table of comparison between K4PCS and our algorithm

For an approximate equivalent time, our algorithm outperforms K4PCS in terms of number of matches. Moreover, we do not need an estimate of the overlap between scans, which prevents bias from a wrong estimation.

5. Conclusion

In this article, we have presented a coarse registration algorithm to address the following challenges: very large

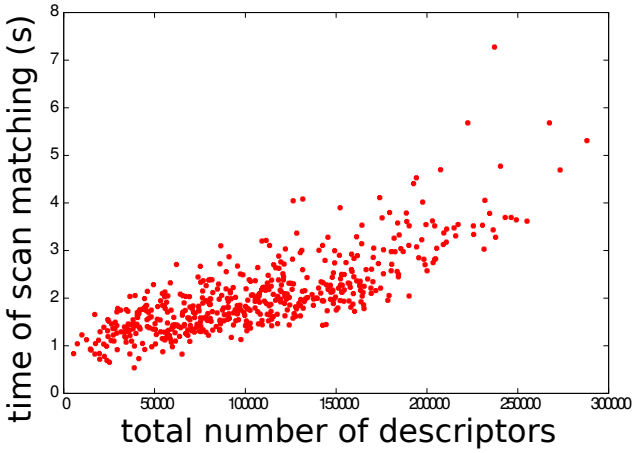

Figure 9. Matching time with respect to the number of descriptors (sum of descriptors in both scans) for each of the 528 pairs of scans. Time of computation increases with number of descriptors.

scale scans (about 10 millions of points), matching pairs of scan with an overlap of 5%, without any guess on the initial alignment, and without the use of any artificial marker nor external sensors. Our algorithm is keypoint-based Point Pair Features and provides 250 correct matches on a database of 33 scans.

In order to evaluate our algorithm, we have acquired a dataset of 33 indoor scans with architectural challenges. To the best of our knowledge, this dataset is the first one publicly available gathering all the aforementioned challenges: large scale, overlap, viewpoints...

We will further improve and increase this database to make it a reference benchmark.

The next step will be to integrate corners since they could for instance be described using the several normals that form it. The detection is still sensitive to occlusions, and this impacts the efficiency of the algorithm as a whole. We are currently exploring an adaptive thresholding strategy on points density of buckets, to lessen the sensitivity of our algorithm to occlusions.

Finally, we are working on a tensor voting approach in order to improve the validation step.

6. Acknowledgments

This work has been partially supported by European Union Horizon 2020 DigiArt project under grant agreement No 665066. (The Internet Of Historical Things And Building New 3D Cultural Worlds, http://digiart-project.eu/).

References

[1] D. Aiger, N. J. Mitra, and D. Cohen-Or. 4 points congruent sets for robust pairwise surface registration. In *ACM SIGGRAPH 2008*, pp 85:1–85:10, NY, USA. 3

[2] H. Badino, D. Huber, Y. Park, and T. Kanade. Fast and accurate computation of surface normals from range images. In *Int. Conf. on Robotics and Automation (ICRA)*, Shanghai, China, 2011. 4

[3] P. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence,*, 14(2):239–256, Feb 1992. 1, 2

[4] T. Birdal and S. Ilic. Point pair features based object detection and pose estimation revisited. In *IEEE 3D Vision*, pp 527–535, 2015. 3

[5] M. Bueno, H. Gonzalez-Jorge, J. Martinez-Sanchez, and H. Lorenzo. Automatic point cloud coarse registration using geometric keypoint descriptors for indoor scenes. *Automation in Construction*, 81:134–148, 2017. 3

[6] Y. Díez, F. Roure, X. Lladó, and J. Salvi. A qualitative review on 3d coarse registration methods. *ACM Comput. Surv.*, 47(3):45:1–45:36, feb 2015. 2

[7] B. Drost, M. Ulrich, N. Navab, and S. Ilic. Model globally, match locally: Efficient and robust 3d object recognition. In *IEEE CVPR*, pp 998–1005, 2010. 3, 4

[8] Y. Guo, F. Sohel, M. Bennamoun, J. Wan, and M. Lu. Rops: A local feature descriptor for 3d rigid objects based on rotational projection statistics. In *Communications, Signal Processing, and their Applications (ICCSPA)*, pp 1–6, 2013. 2

[9] A. Makadia, A. I. Patterson, and K. Daniilidis. Fully automatic registration of 3d point clouds. In *IEEE CVPR*, pp 1297–1304, Washington, DC, USA, 2006. 2

[10] N. Mellado, D. Aiger, and N. J. Mitra. Super 4pcs fast global pointcloud registration via smart indexing. *Computer Graphics Forum*, 33(5):205–215, 2014. 3

[11] M. Mohamad, D. Rappaport, and M. Greenspan. Generalized 4-points congruent sets for 3d registration. In *IEEE Int. Conf. on 3D Vision*, pp 83–90, USA, 2014. 3

[12] A. Nuchter, K. Lingemann, and J. Hertzberg. Cached k-d tree search for ICP algorithms. In *Int. Conf. on 3-D Digital Imaging and Modeling*, pp 419–426, 2007. 2

[13] M. Pauly, M. Gross, and L. Kobbelt. Efficient simplification of point-sampled surfaces. In *IEEE Visualization*, pp 163–170, Nov 2002. 5

[14] S. Rusinkiewicz and M. Levoy. Efficient variants of the icp algorithm. In *Int. Conf. on 3D Digital Imaging and Modeling*, pp 145–152, 2001. 2

[15] I. Sipiran and B. Bustos. Harris 3d: A robust extension of the harris operator for interest point detection on 3d meshes. *Vis. Comput.*, 27(11):963–976, nov 2011. 2

[16] P. W. Theiler, J. D. Wegner, and K. Schindler. Keypoint-based 4-points congruent sets automated marker-less registration of laser scans. *ISPRS Journal of Photogrammetry and Remote Sensing*, 96:149–163, 2014. 3

[17] A. Zaharescu, E. Boyer, K. Varanasi, and R. Horaud. Surface feature detection and description with applications to mesh matching. In *IEEE CVPR*, pages 373–380, 2009. 2

[18] B. Zeisl, K. Köser, and M. Pollefeys. Automatic registration of RGB-D scans via salient directions. In *IEEE ICCV*, Sydney, Australia, pp 2808–2815, 2013. 3

[19] Z. Zhang, S. H. Ong, and K. Foong. Improved spin images for 3d surface matching using signed angles. In *IEEE ICIP*, pp 537–540, 2012. 2