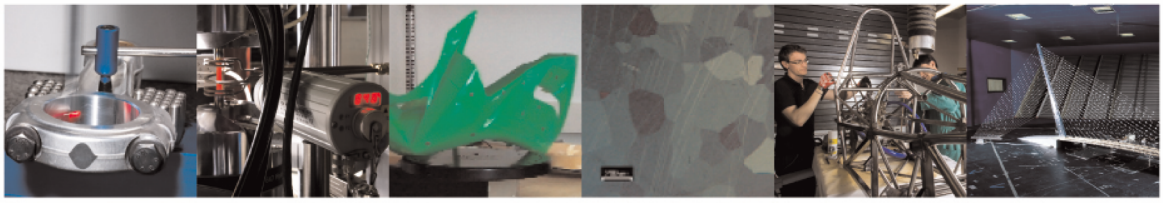
POLITECNICO
MILANO 1863

DIPARTIMENTO DI MECCANICA

mecc



Virtual simulation benchmark for the evaluation of simultaneous localization and mapping and 3D reconstruction algorithm uncertainty

Daniele Marchisotti and Emanuele Zappa

This is the Accepted Manuscript version of an article accepted for publication in Measurement Science and Technology. IOP Publishing Ltd is not responsible for any errors or omissions in this version of the manuscript or any version derived from it. The Version of Record is available online at *https://dx.doi.org/10.1088/1361-6501/abeccc*

DIPARTIMENTO DI MECCANICA ■ POLITECNICO DI MILANO
via G. La Masa, 1 ■ 20156 Milano ■ EMAIL (PEC): pecmecc@cert.polimi.it
http://www.mecc.polimi.it
Rev. 0

Virtual simulation benchmark for the evaluation of SLAM and 3D reconstruction algorithms uncertainty

Daniele Marchisotti, Emanuele Zappa

[1] Department of Mechanical Engineering, Politecnico di Milano, Via La Masa 1, Milan, Italy
[2] Department of Mechanical Engineering, Politecnico di Milano, Via La Masa 1, Milan, Italy

E-mail: daniele.marchisotti@polimi.it, emanuele.zappa@polim.it

Received xxxxxx
Accepted for publication xxxxxx
Published xxxxxx

Abstract

Simultaneous Localization And Mapping (SLAM) algorithms allow to obtain a unique 3D shape and 3D sensor trajectory by combining partial scans obtained moving a 3D scanner. Performances of these algorithms are significantly affected by experimental conditions, characteristics of the target and values of the parameters of the reconstruction algorithm. Therefore, uncertainty and reliability of SLAM techniques need to be assessed before their application, e.g. for robot navigation, for autonomous vehicles or industrial fields. To evaluate the uncertainty of these algorithms, specific datasets containing 3D scans, with the possibility to control different conditions, e.g. sensor trajectory, depth or color noise, sensor velocity and framerate, are necessary. In this article, we present a procedure to obtain virtual datasets with the complete control on environment, 3D sensor and trajectory conditions, starting from any real 3D dataset acquisition, characterized by a sufficiently low uncertainty. These datasets could be generated to test the effect of SLAM algorithm parameters, to determine the best parameters to be used to exploit the algorithm characteristics to obtain the best result in each operating context. The advantage of this procedure is the possibility to perfectly control each condition and to evaluate its effect on the final result. This procedure was applied to two reconstruction algorithms, as examples, namely: Open3D reconstruction tool and ElasticFusion. Results demonstrate that the setting of algorithm parameters, e.g. the tolerance on depth correspondence between frames or the number of fragments, or the change of number of frames acquired, can have a strong influence on the resulting 3D reconstruction and trajectory. Moreover, the effect of not closing the loop trajectory on reconstruction performances is quantified for different application scenarios.

Keywords: 3D reconstruction, virtual datasets, odometry, loop closure, Open3D, RGBD, SLAM

1. Introduction

The possibility to create datasets and benchmarks adapted to different tests and scenes can improve scientific evaluation metrics in the classification of algorithms, in terms of performances of 3D reconstruction systems. Indeed, the presence of open source datasets and clear evaluation techniques can represent a standard metric to measure 3D

reconstruction algorithm results. For example, the RGB-SLAM dataset provided by Sturm et al [1] has been used extensively to test many different reconstruction algorithms [2],[3],[4]. This dataset is based on the acquisition of RGBD frames from Kinect sensor and on a ground-truth sensor trajectory estimated by a high accuracy motion-capture system equipped with high-speed tracking cameras. This type of datasets is particularly useful for SLAM (Simultaneous Localization And Mapping) systems, where the outputs are the 3D scene reconstructed and the camera trajectory, while the input are RGBD frames, given by a RGBD sensor, as a Structured Light, Time Of Flight or stereo camera [5]. These datasets could be used also for 3D reconstruction systems, where the main goal is to obtain a precise 3D map and, secondary, the sensor trajectory. However, many datasets are limited to the conditions of the ambient where they are acquired, e.g. texture, lighting, motion blur and so on, as well as the frame rate, starting point and sensor motion speed. In addition, testing a new SLAM or visual odometry algorithm could require many different dataset acquisition conditions to evaluate specific critical situations. Moreover, validating an algorithm in the many different operating conditions of RGBD frame sequences (e.g. on different overlapping percentage between consecutive and not consecutive frames) could give more information about the goodness of the final reconstruction.

In general, SLAM algorithms are based on features detection performed on RGB and depth images and on matching among frames to recreate a 3D map and a sensor trajectory [6]. Two different computation steps are generally included into SLAM algorithms: visual odometry and loop closure [7]. The difference between the two steps is that visual odometry is an ego-motion technique and the trajectory sensor is obtained from the detection of features in consecutive frames, while for loop closure the features of non-consecutive frames are matched to determine previously-visited locations by the sensor. These algorithms can be combined with data acquired by on-board sensors, as IMU and GPS [8][9]. Other techniques, as Structure From Motion combined with Kalman Filter, have also appeared [10]. In the Open3D, as well as in other reconstruction systems, it is possible to tune the system to rely more on visual odometry than loop closure and the opposite. The impact on the final result of one technique with respect to the other must be set in a correct way to obtain satisfactory results.

Testing 3D reconstruction and SLAM algorithms usually require several different datasets to study the effectiveness of an algorithm under different sensor and ambient conditions. SLAM algorithms conventionally refer to online estimation, but for testing they can be used offline by providing RGBD frames sequence in a set of files, including timestamp information. In this article, we present a procedure for testing SLAM and 3D reconstruction algorithms. This method is based on a virtual 3D scanner, identified by sampling a colored point cloud to simulate the measurements obtained by a RGBD sensor translating or rotating in the point cloud space. By shifting or rotating the virtual sensor with respect to the point cloud, it is possible to generate a sequence of RGBD frames with associated scanning trajectory, which can be used as a ground truth for a comparison with the trajectory estimated with SLAM algorithms.

The ability to create artificially a dataset from a real high-precision 3D point cloud can be really useful to test many conditions as the ones mentioned before, e.g. it is possible to apply disturbances on the acquired point clouds to simulate real sensors and environmental conditions. A first example of application of the presented test procedure is referred to the Open3D reconstruction system [11],[12], to spot the changes in the 3D reconstruction results (3D map and sensor trajectory) with respect to the disturbances and the change of parameters of the reconstruction system itself. In this work, the textured point clouds to be sampled with the virtual scanner are the ones available in the dataset, used as ground truth in [11] and acquired with FARO Focus 3D X330 HDR scanner, that has an accuracy of 0.1 mm at a distance of 10 m and a measurement range from 0.6 to 330 m. The scans, obtained directly from the sensor, were merged using dedicated software provided by the manufacturer and aligned with a program provided, as described in [11]. Although the analysis in this work are mainly based on the mentioned dataset and with Open3D reconstruction tool, in the sections 7-8 the proposed procedure is applied to the RGBDObject dataset [13] and the reconstruction is obtained with ElasticFusion [14] approach to show the generality of this evolution technique. These two algorithms were chosen since they are two state-of-the-art algorithms with deeply different approches. While Open3D can be used only for post-processing, ElasticFusion can also be used online during the acquisition with a sensor. Open3D relies on the optimization of a pose graph to obtain a correct reconstruction, while ElasticFusion is based on a deformation graph that is used for non-rigidly deforming surfels according to surface constraints provided loop closure methods [11][14]. This deformation is applied by considering the effect of a set of influencing nodes. Open3D divides a RGBD sequence in sets of images to compute fragments, while this cannot be done for ElasticFusion, which does not operate in post-processing.

This testing benchmark can be useful to find an optimal combination of the parameters of SLAM or 3D reconstruction algorithms before performing experiments. In this case, by performing simulations by changing the algorithm parameters, a set of parameters can be derived to obtain the best results for an algorithm in specific conditions. However, during the SLAM or 3D reconstruction process, it is possible to change the algorithm parameters if the algorithm allows to change them, during processing.

The RGBD sequences extracted with this procedure are used as a reference for SLAM and 3D reconstruction algorithms testing. The aim of this work is to study the final results of algorithms by changing the ambient conditions. If it is necessary to study the effect of a specific disturbance on an algorithm, only that disturbance will be applied on the dataset.

2. SLAM algorithms testing: related works

In scientific literature, there are open-source systems for the simulation of autonomous vehicles navigation (cars, drones and robots) [15]. A similar approach is applied to robotics [16]. In general, these two simulation environments are used for testing autonomous navigation and they are not focused on a deep metrologial analysis. Another simulation framework is the one presented in [17], where datasets are generated from artificially-created environments. However, these simulation environments are based on computer models of environments, where the simulation is performed, while our system starts from sampling a point cloud extracted from data acquisition of a real scene. For this reason, our system is more similar to a real acquisition system, with a higher level of detail in terms of simulation for metrological analysis. This can be an advantage with respect to a simulator that uses an artificially-created environment.

After generating a new dataset with the desired conditions, a SLAM algorithm can be tested and the result is a 3D map given by data acquired from the sensor and an estimated trajectory, described by a pose graph. These 2 outputs can be used to identify the ability of the SLAM algorithm to reconstruct a 3D scene and to be applied to robot and vehicles navigation.

In literature, there are several approaches to evaluate uncertainty of reconstruction trajectories and 3D maps. Many of those are related to a comparison to a ground truth scene acquired with a more precise sensor or CAD model or a comparison to a ground truth trajectory obtained from a high-accuracy motion capture system.

2.1 Indicators for point cloud comparison

For point clouds comparison, the simplest and most intuitive way is to compute the distance between each point of the reconstructed point cloud and its closest one in the ground truth point cloud and the opposite (ground-truth and reconstructed point clouds could not have the same number of points). However, this approach is prone to the presence of outliers, that can highly increase these indicators and lead to misleading results. For this reason, more stable techniques were developed as the nearest neighbor distance with Hausdorff distance [18], where two subsets of reconstructed and ground truth point clouds are taken and the distance computed is the greatest of all the distances from a point in one set to the closest point in the other set. This approach is very suitable in the case of a low-density ground truth point cloud. Another approach can be to fit small neighbors of a point cloud with planes with least square, triangulation or quadratic functions [19] and compute distances as Euclidean or L1-L2 norms [20]. A new approach, suggested in [11] is to establish a threshold for the distance between 2 closest points belonging to 2 different point clouds to determine a correspondence. Among the mentionted indicators, in this work we focus on *precision* and *recall*, to compare the reconstructed 3D cloud with the ground truth one, since they allow a concise and intuitive index of reconstrution accuracy.

1. Precision: the percentage of reconstructed points that have a ground-truth point within the threshold distance. Wrong parts of reconstruction point cloud reduce precision;
2. Recall: the percentage of ground-truth points that have a reconstructed point within the threshold distance. Incomplete reconstruction reduce recall;

In [11], it is suggested as distance threshold τ=20 mm, for 3D reconstruction of environments with dimensions of few meters, i.e. the same threshold it was used in this paper (for environments or objects of smaller dimensions, it is recommended to set a smaller value for τ). The 3D scanner uncertainty can influence the choice of the threshold, as well (a larger sensor uncertainty corresponds to a higher value of τ).

2.2 Indicators for trajectory evaluation

For sensor trajectory evaluation, the most used approach is to make a comparison between the ground truth trajectory and the reconstructed one. This comparison can be obtained by extracting the Euler angles and 3D space coordinates from pose graph nodes of both ground-truth and reconstructed trajectories. However, this comparison is made with 6 variables, thus it is not compact, although it can be used to spot specific errors in case of drift of single variables. More general indicators, that are widely used and applied in this work are:

1. The Absolute Trajectory Error (ATE) [1]: the Euclidean distances between all estimated camera poses and the ground truth poses and measures the global accuracy of the estimated trajectory;
2. The Relative Pose Error (RPE) [1]: it measures the local accuracy of a trajectory over a fixed time interval, i.e. the measure of drift in a trajectory.

These indicators were used in [2],[3],[4]. For our datasets, since the pose of the sensor is known because it is imposed virtually, the comparison between reconstruction and ground truth can be made for both trajectories and 3D maps.

3. SLAM and 3D reconstruction simulation method

Data acquisition with RGBD sensors is often influenced by sensor random and systematic errors and by environmental conditions, such as lighting, motion blur, scene texture and so on. These disturbances are very difficult to exclude in post-

processing of frames. Our dataset is generated virtually from a colored point cloud acquired with high accuracy Faro 3D scanner obtained from [11], by simulating a RGBD sensor sampling the colored point cloud. In our case, the ambient conditions are the same for all datasets generated from the same point cloud and possible introduction of disturbances is fully controlled when the dataset is generated. The 3D scenes for our experiment consist of 2 indoor room scenes of a flat, as shown in fig. 1 (scenes 1-2). Furthermore, point clouds from RGB-D Object Dataset [13] were used to obtain other 3 scenes and they are introduced in section 7.

Fig. 1. 3D scenes used for our experiment [11]: scene 1 (top), scene 2 (bottom). The scene 2 was split in 2 parts to show the internal part of the room

The output of SLAM simulation is a series of synchronized RGB and depth images, that are extracted in our system by following these steps:

1. Identification of RGBD intrinsic parameters: image width and height (w, h), focal length (fx, fy), FOV (Fx, Fy) and optical center (cx, cy). These parameters are the same for RGB and depth images (the origin of RGB and depth images is the same);

2. RBGD camera origin placement: the virtual sensor is placed at a distance not too close and not too far to the scene that is framed. In the first case, this is done to have enough points within the region of interest and for each 3D sensor pixel and, in the second case, to avoid to simulate a condition that is not realistic respect to the maximum distance that can be measured by common sensors available on the market;

3. Sensor trajectory choice: the trajectory is completely defined by a sequence of transformation matrices. The transformations are referred to the local sensor reference system;

4. Point cloud preprocessing: denoise the original point cloud to remove outliers and reduce imperfections [21](eventually, down-sample or up-sample [22] algorithms can be considered to obtain the point cloud density required);

5. Point cloud sampling: the algorithm sequence is visible in the chart below. The output of this step is the sequence of RGBD frames, representing the SLAM simulation;

Algorithm 1 Point Cloud Sampling

Input: Pre-processed point cloud **P** centred in the sensor origin, sensor poses defined by a sequence of transformations $\mathbf{T_s}$ with respect to the origin of the trajectory

Output: RGBD frames sequence for SLAM simulation

1: ***for*** $\mathbf{t} \in \mathbf{T_s}$ do

2: Get points coordinates $\mathbf{P_c}$ of point cloud **P**

3: Apply transformation: $\mathbf{P_c}' = \mathbf{P_c} * \mathbf{t}$

4: Consider only points inside the camera FOV, i.e. respecting the following conditions:

$$\begin{cases} P_{c,x}' < \tan\left(\frac{F_x}{2}\right) \\ P_{c,x}' > -\tan\left(\frac{F_x}{2}\right) \\ P_{c,y}' < \tan\left(\frac{F_y}{2}\right) \\ P_{c,y}' > -\tan\left(\frac{F_y}{2}\right) \end{cases}$$

5: Divide the camera horizontal and vertical FOV in a number of subspaces equal to the width and height of the camera image. Each subspace $S_{i,j}$ for $i = 1,2,\dots,w, j = 1,2,\dots,h$ is defined by the following constrains:

$$S_{i,j}: \begin{cases} s_{(i,j),x} < \tan\left(-\frac{F_x}{2} + \frac{F_x}{w} * i\right) \\ s_{(i,j),x} > \tan\left(-\frac{F_x}{2} + \frac{F_x}{w} * (i-1)\right) \\ s_{(i,j),y} < \tan\left(-\frac{F_y}{2} + \frac{F_y}{h} * j\right) \\ s_{(i,j),y} > \tan\left(-\frac{F_y}{2} + \frac{F_y}{h} * (j-1)\right) \end{cases}$$

6: Divide the points $\mathbf{P_c}'$ into each subspace $S_{i,j}$ obtaining $\{P_c'\}_{i,j}$

7: For each $\{P_c'\}_{i,j}$, select the point $P_{i,j}{}'$ closest to the pixel line $l_{i,j}$ given by the equations for $i = 0,1,\dots,w-1, j = 0,1,\dots,h-1$:

$$l_{i,j}: \begin{cases} l_{(i,j),x} = \tan(-\frac{F_x}{2} * \left(1 - \frac{1}{w}\right) + \frac{F_x}{w} * i) * l_{(i,j),z} \\ l_{(i,j),y} = \tan(-\frac{F_y}{2} * \left(1 - \frac{1}{h}\right) + \frac{F_y}{h} * j) * l_{(i,j),z} \end{cases}$$

8: Save the point Z coordinate $P_{i,j,z}{}'$ in the pixel (i,j) in the depth image and the point RGB information $P_{i,j,RGB}{}'$ in the pixel (i,j) in the color image

---

6. RGBD frame postprocessing: in case no points are present in the subset $\{P_c'\}_{i,j}$, the result is a null pixel in the RGB and depth images. These points can be

filtered by analysing image region properties and by applying a median filter to those null "particles" (median filter was applied to reduce the influence of outliers).

To apply the described procedure, it is necessary to set few parameters, e.g. camera intrinsic parameters. For our experiments, the following parameters were set:

1. Camera intrinsic parameters:
   a) Image width: w=640 pixels;
   b) Image height: h=480 pixels;
   c) Focal length: fx=618.30 px, fy=618.13 px pixels;
   d) Optical centre: cx=320 px, cy=240 px;
   e) FOV: $F_x = 2 * \arctan\left(\frac{w}{2*f_x}\right) = 54.7°, F_y = 2 * \arctan\left(\frac{h}{2*f_y}\right) = 42.4°$.

   These intrinsic parameters are the same of Intel RealSense D145 camera with the image width and height specified.
2. The maximum distance of the sensor from the framed scene is about 5.2 m and 3.7 m for scenes 1 and 2 and it is about half the maximum measurable distance of Intel RealSense D415 sensor, while the minimum distance is about 0.5 m, larger than the minimum sensor measurable distance (0.2-0.3m) [23];
3. The trajectory of the sensor is chosen to analyse separately the effect of translation from the rotation of the sensor. Therefore, a translation-only circular trajectory of radius of 0.5 m is defined for the scene 1 in fig. 1, while a 360° rotation of the sensor around the Y axis (vertical axis) of the camera for the scene 2 in fig. 1. The trajectory loop is closed in both cases. For the 3 scenes obtained from the RGB-D Object Dataset, an orbit trajectory for the sensor was defined. This trajectory was determined to make the sensor navigate around a scene and look at the centre of it;
4. For our experiment, outlier removal is identified by a threshold, which determines that a point is considered to be an outlier, if the average distance to its k-nearest neighbours is above the specified threshold. In our case, the threshold is 0.08 m;
5. The median filter size to remove null blobs in RGB and depth images, for our scenes, is fixed to 20x20 px. These values were set to remove those small null "particles", in case of absence of points in the point cloud to sample for those pixels. The median filter is applied only to null particles.

After generating the sequence of RGBD frames, a new ground truth point cloud is generated by aligning the frames generated with the ground truth trajectory. The resulting point cloud is used to calculate precision and recall indicators.

The resolution of depth pixels is 1 mm, while the resolution of X and Y coordinates of points respect to the sensor reference system are 0.32-8.42 mm for a depth range of 0.3-5.2 m with the selected sensor focal length and resolutions.

With respect to the virtual sensor reference system, this procedure of creating datasets by sampling a point cloud has an uncertainty due to the resolution of the sensor. According to [24], the corresponding uncertainty is equal to the resolution divided by $2\sqrt{3}$ and, for the resolutions written in the previous paragraph, the uncertainty is 0.29 mm for depth and of 0.09-2.44 mm for X and Y for a depth range of 0.3-5.2m and with the selected sensor characteristics (95% confidence level). There is no uncertainty regarding the sensor trajectory, since it is virtually determined with no connection to the point cloud to sample.

The maximum uncertainty of ground truth point cloud depends on the sensor to be simulated. Usually, in sensor static calibration, the reference must have an uncertainty an order of magnitude lower that the sensor to be calibrated. In our case, since the ground truth point cloud has an uncertainty of 0.1 mm, it can be used for the simulation of a sensor with depth resolution of 1 mm.

Since the point cloud to calculate precision is generated by using the sequence of RGBD images, the uncertainty of the point cloud to generate the RGBD sequence does not affect the result in terms of precision and recall.

The minimum density required of the ground truth point cloud depends on the distance from the scene to acquire at which the simulation is performed and on the simulation sensor resolution, since with a high resolution and a low distance there would be no points in the FOV of a single pixel. At least one point should be present in the FOV of a single pixel to avoid "holes" in the resulting dataset. For example, supposing to acquire a flat surface perpendicular to the sensor at a distance of d = 1 m, the minimum point cloud density along x and y directions (D_x, D_y) would be:

$$D_x = \frac{1}{\frac{2*\tan\left(\frac{F_x}{2}\right)}{w}*d}, D_y = \frac{1}{\frac{2*\tan\left(\frac{F_y}{2}\right)}{h}*d} \quad (1)$$

For the tested sensor, since the pixels are squared, $D_x = D_y = 619\ \text{points/m}$. This point cloud density is defined as $D = D_x * D_y = 383161\ points/m^2$, for acquiring at a distance of 1 m. In case of lower point cloud density in small parts of the acquisition, a median filter of images can be applied, as previously explained.

With this simulation system, it is possible to create datasets simulating all different conditions in a controlled way. Indeed, disturbances with specific standard deviations or a different number of frames for the same trajectory are parameters that can be relevant for the final result. The scope of this work is to propose a technique to evaluate 3D reconstruction algorithms in all their aspects by isolating single effects on the final result.

4. Open3D reconstruction system

As a case study to apply our algorithm testing method, we have considered a state-of-the-art 3D reconstruction system, that is part of one of the most recent point cloud processing libraries: Open3D. According to the documentation provided for the reconstruction system in Open3D [11], [25], this system is based on 4 phases:

1. Making fragments: building local geometric surfaces (fragments) from short sub sequences of the input RGBD sequence. Local registration is mainly performed by using visual odometry and with a small contribution for loop closure (this small contribution is determined by the parameter preference loop closure odometry, set by default to 0.1);
2. Registering fragments: performing global registration and detecting fragment pairs that can be matched by applying loop closure over fragments. Loop closure contribution is very high, with preference loop closure registration parameter set to 5;
3. Fragments registration refinement: more tight registration of fragments by using colour ICP and multiway registration. The algorithm is described in detail in [9]. This process consists of optimizing pose graph by minimizing a colour and geometric cost function;
4. Scene integration: 3D scene reconstruction with Truncated Signed Distance Function (TSDF) volume integration [26]. TSDF frames integration works like weighted average filter in 3D space. If more frames are integrated, the volume will produce smoother point cloud.

The Open3D reconstruction system can be configured by setting few parameters, related to the type of scene and data recorded. Few of these parameters are the following (the other parameters were set to default values of the algorithm):

1. max_depth: maximum depth value threshold for RGBD frames (set to 7m for our experiments presented in section 6);
2. min_depth: minimum depth value threshold for RGBD frames (set to 0.3m for our experiment, suitable to simulate a real depth camera, as Intel RealSense D415);
3. max_depth_diff: maximum depth distance for each pixel correspondence between different frames (this parameter was changed in our experiment to quantify its effect in the final result);
4. preference_loop_closure_odometry: parameter to determine the odometry vs. loop closure trade-off for the alignment of frames inside a fragment. It is a parameter between 0 and 1 (set to 0.1, meaning high relevance for odometry during pose graph optimization within a fragment);
5. preference_loop_closure_registration: parameter to determine the odometry vs. loop closure trade-off for fragments registration (set to 5 to make the system align fragments mainly with loop closure);
6. n_frames_per_fragment: number of frames in a fragment. This parameter was changed many times during our experiment to change the number of fragments and, consequently, the larger or lower effect of loop closure respect to odometry.

An example of the reconstructed trajectories for scenes 1-2 are visible in fig. 2-3, respectively (the reference system of the sensor is shown, as well). For scene 1, it is clearly observable the circular trajectory on the X-Z plane referred to the sensor reference system. For the scene 2 (fig. 3), involving a rotation-only trajectory around the Y vertical axis of the camera, the reconstruction result can be expressed in terms of rotations with respect to the 3 axes. The rotations around the axes X, Y, Z are the rotations to apply to each frame to align the reference system of each frame with respect to the initial reference system. These rotations can be seen as rotation angles in the rotation matrix $R = R_x * R_y * R_z$. They are not representing the change in direction of each axis. As it can be observed, the rotation around the Y axis changes its versus and at the same time there is a rotation shift of 180° around X and Z axes.

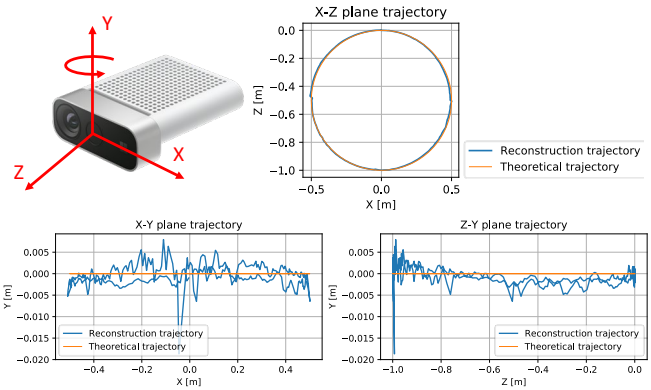


Fig. 2. A scheme of the sensor reference system (the rotation around the Y axis is shown for scene 2) and an example of trajectory reconstruction in terms of X,Y,Z displacements for scene 1

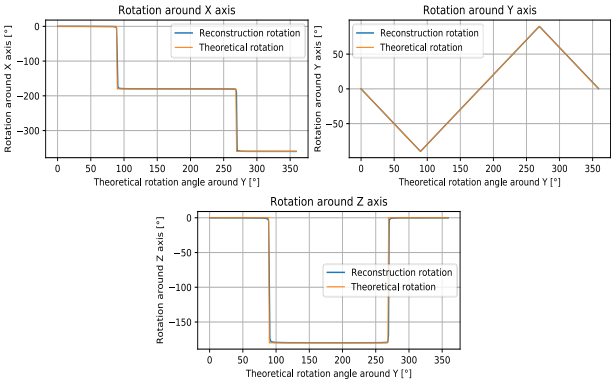


Fig. 3. Example of trajectory reconstruction in terms of rotation angles with respect to the 3 axes for scene 2 (the rotations are related to the reference system in fig. 2)

5. Algorithm test method

As stated in the previous section, testing a SLAM algorithm can include tests on datasets acquired from different sensors, for different ambient conditions and acquisition conditions. For this reason, it is necessary to identify the robustness of the system to higher or lower framerate with respect to sensor translation and rotation velocity. To determine the effect of each aforementioned condition on the result of the tested SLAM algorithm, the Open3D reconstruction system was tested under the following conditions:

1. Combination of depth and colour random noise: application of noise on depth and colour images with gaussian distribution of different standard deviations. A new dataset was generated for each condition;
2. Different max_depth_diff values: parameter of the reconstruction system, tested with respect to the depth random error;
3. Combination of depth gaussian noise and colour gaussian filter: application of gaussian noise on depth images and gaussian filter on RGB images, with different standard deviations;
4. Number of frames and fragments: changing the number of frames of the total RGBD sequence and the number of fragments;
5. Incomplete trajectory: excluding few frames at the sequence end in order not to close the trajectory loop. The effects of changing the number of frames and fragments is investigated also for these new trajectories to identify possible changes with respect to complete trajectories.

Tests to be performed can change depending on the algorithm parameters, while for noises this procedure can be adopted for different SLAM algorithms to make comparisons and determine the robustness of these algorithms.

The evaluation of the final reconstruction result is determined by comparing the ground-truth and the 3D reconstructed trajectories. For both the scenes 1 and 2, it was computed the ATE and the RPE index. In [1], the ATE index is evaluated after translating and rotating the ground-truth trajectory with the Horn method [27] to align it with the one reconstructed with a rigid-body transformation found through a least-squared solution. This alignment is necessary when the two trajectories are represented in two different reference systems. In our experiment, both trajectories are from the same reference and, therefore, this step is not required. In our case, the ATE is coincident with the absolute error summed over the 3 spatial coordinates. The ATE and RPE are extracted for each camera pose of the trajectory and the RMSE of these indicators is computed. For an evaluation of the global consistency of the trajectory, only the ATE is used for the evaluation of trajectories uncertainty, while the RPE is more suitable for local accuracy. For this reason, only the ATE is used in this article, to evaluate algorithm performances.

The estimation of the correspondence between ground-truth and reconstructed 3D scenes is assessed by computing precision and recall, as defined in [11]. However, the precision of the 3D reconstruction is the only one that we have used in this article, since the recall would be affected by the presence of parts of the 3D environment that were not framed.

6. Results and discussion

A first attempt of the application of the procedure explained in the section 3 is the evaluation of Open3D reconstruction system performances. In particular, our study focused mainly on testing the robustness of the algorithm with respect to different disturbances, the effect of algorithm parameters and the effect of changing the number of frames in the sequence. These factors are tested on both scenes 1 and 2.

6.1 Depth gaussian noise and max_depth_diff (MDD)

The max_depth_diff parameter is the maximum depth distance between two pixels p_i, q_j of different point clouds of the same fragment to be considered as correspondent points. This parameter should be adapted to the sensor noise level and to the velocity of the RGBD camera, in order to obtain consecutive frames with a depth difference not larger than this parameter. For this reason, the robustness of the system with respect to depth gaussian noise is derived by testing different MDD values for different depth frame random error levels. For this test, the number of fragments was set to 4 and the number of frames to 360. Fig. 4-5 show this correspondence for scenes 1 and 2.

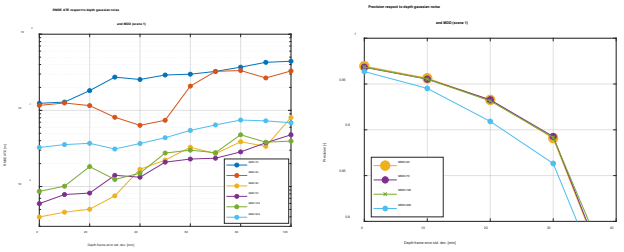


Fig. 4. RMSE ATE and precision for different MDD and depth random noise values for scene 1

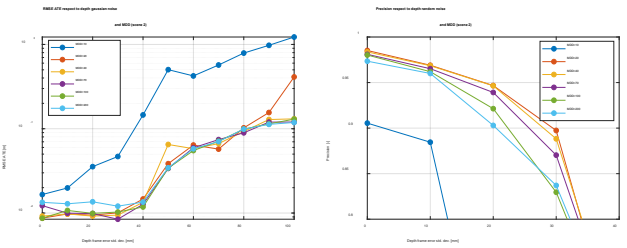


Fig. 5. RMSE ATE and precision for different MDD and depth random noise values for scene 2

Fig. 4 (left) shows that the lowest RMSE ATE errors are obtained for MDD equal to 40 mm and 70 mm. Indeed, for lower values of MDD, many possible corresponding points

could be not considered for aligning frames, since the depth difference is larger than MDD. On the contrary, with too high values of MDD (100 and 200 mm), wrong correspondences of points with large depth difference could be included, leading to a reduction of performances. Moreover, with MDD=200 mm, the precision decreases, as well (fig. 4, right). For scene 2 (fig. 5), there is a different effect of MDD, but the trend is similar. Setting the parameter to 10 mm, the RMSE is higher respect to larger values of MDD and precision has the highest values for MDD equal to 40 mm. For both scenes, the reduction of precision is highly affected by the depth gaussian noise. In fact, for values of depth gaussian error larger than 40 mm, the precision is lower than 0.8.

Previous considerations demonstrate that an optimal value for MDD parameter can be found, as a trade-off between too low values leading to an insufficient number of correspondences between frames and too high values with the presence of wrong correspondences. Indeed, low values are suitable for low noise and slow-motion datasets and high values should be set for fast camera motion and high depth noise sensors. For example, for Intel RealSense D415 camera, that has a standard deviation depth random error of about 10 mm at 1.5 m depth [23] for all pixels, the MDD parameter could range from 40 to 100 mm, depending on the sensor speed, during acquisition.

Furthermore, it can be noticed that precision of the 3D scene is more sensitive to changes of MDD parameter for a rotation-only trajectory (scene 2) respect to a translation trajectory (scene 1). This is likely to be caused by the variation of depth at the border of the depth image between corresponding pixels is larger for a rotation trajectory respect to a translation trajectory. Moreover, the reduction of precision is also connected to the integration of frames in the final 3D map. For a large noise level (above 40 mm of standard deviation) of dataset frames, there is a larger noise level in the final 3D reconstructed scene.

Relying on the results discussed above, for all the other 3D reconstruction tests performed in this article, the MDD parameter was set to 70 mm.

6.2 Depth gaussian noise and colour gaussian noise/colour gaussian filter

Features derived from colour and depth images are essential to align all frames in a sequence, for RGBD visual odometry and loop closure. Thus, the application of gaussian noise on colour images, combined with depth gaussian noise, could reduce the accuracy of the final result. Two different types of tests were performed: one regarding the application of depth and colour gaussian random noise and one regarding the application of depth gaussian noise and colour gaussian filter. With respect to results presented in the section 6.1, it was observed that depth noise has a more relevant effect respect to colour noise. Fig. 6-7 show results of these two tests, respectively. As shown in fig. 6, the colour gaussian noise has negligible effect on the absolute mean error of the 3D reconstruction, even when high levels of colour random noise are applied to dataset frames (standard deviation equal to 10% of the full scale represent real noise in case of very bad environmental conditions). The trend of the precision with respect to random noise variations is not shown, since it is analogous to the trend of RMSE ATE.

The application of a gaussian filter to colour images (fig. 7) can increase the ATE in a relevant way when the standard deviation of the gaussian filter is higher than 6 pixels, corresponding to about 18-51 mm for distances from the camera ranging between 2 and 5.2 m, respectively. As can be seen in fig. 5-6, the ATE is more affected by colour gaussian noises and colour gaussian filter in the scene 1. This is reasonable, because the depth range for scene 1 is 3-5.2 m, while for scene 2 is 2-3.7 m and the gaussian blur and gaussian noise have a larger effect at higher distances, where a lower number of pixels is used for describing a feature. From results presented in this section and in section 6.a, considering typical levels of standard deviation of depth noise (e.g. 20-30 mm) and colour noise (e.g. 0.3 px) of commercial RGBD sensors, it is possible to observe that the depth gaussian noise has a higher impact on the final result than the colour gaussian noise. The same consideration applies to the colour gaussian filter, with typical values in case of image blur of 2-3 px.

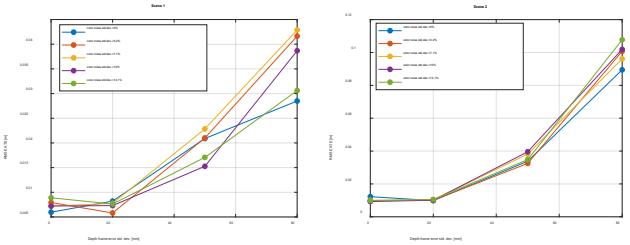


Fig. 6. RMSE ATE for different colour random noise and depth random noise levels for scenes 1 and 2, respectively

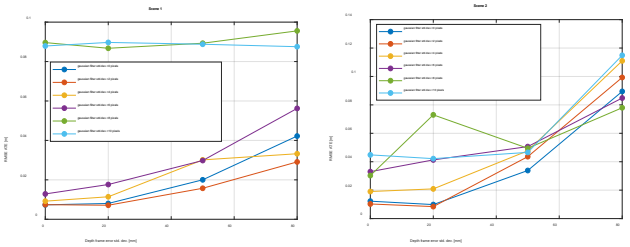


Fig. 7. RMSE ATE for different colour gaussian filter and depth random noise levels for scenes 1 and 2, respectively

6.3 The number of frames and fragments

In real SLAM acquisitions, it can happen that for few parts of the entire 3D scene to be reconstructed, only a low number of frames is acquired and, in the same acquisition, other parts can be defined by a larger number of frames. This can have a high impact on the final result, if this aspect is ignored. In this case, tuning parameters of the reconstruction system can help

to reduce the errors and to obtain a satisfactory result. One of these parameters in Open3D reconstruction system is the number of fragments, since within single fragments the odometry has a high relevance, while alignment of different fragments is mainly performed with loop closure. In the case of a low number of frames, the loop closure can help to adjust possible odometry drifts in a more accurate way, if there are parts in common between non-consecutive frames. On the other side, in case of high number of frames, the camera motion can be determined precisely by odometry, with high overlap between consecutive frames.

To test the algorithm under evaluation, datasets with the same camera trajectory but with a number of frames between 120 and 720 were generated and each of them was tested with a different number of fragments. This number of frames is defined considering that the typical framerate of RGBD sensors is 30 Hz: with a dataset with 720 frames, this would mean an acquisition time of 24 s for a complete rotation or circular translation trajectory. The minimum number of frames was set to avoid to have an effect of MDD (set to 0.07 m) in this test, because in this way the same features in consecutive frames would not have a depth difference larger than 0.07 m. For the 120 frames dataset of scene 2, the percentage of points in subsequent frames which depth difference is larger than 0.07 m is about 10%.

In fig. 8-9, it is shown the RMSE ATE and the precision for scenes 1-2 for a relatively high number of frames in one dataset (between 360 and 720 frames, meaning a rotation difference between 2 frames of 0.5-1° for scene 2 and a translation of 4.36-8.73 mm for circular rotation trajectory with radius of 0.5 m for scene 1).

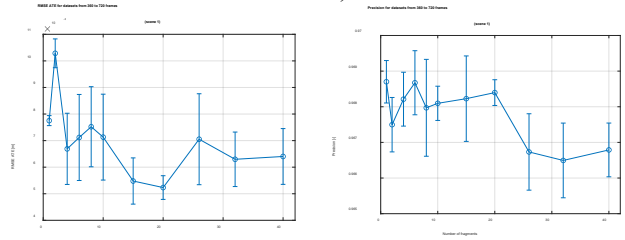


Fig. 8. RMSE ATE and precision for scene 1 for datasets with a number of frames between 360 and 720 respect to the number of fragments

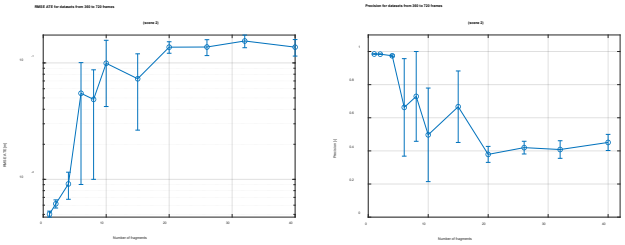


Fig. 9. RMSE ATE and precision for scene 2 for datasets with a number of frames between 360 and 720 respect to the number of fragments

For scene 1 (fig. 8), the difference in the result of the 3D reconstruction and trajectory passing from 1 to 40 fragments is negligible. This is not true for scene 2, where using a low number of fragments (up to 4) can be considered as the best choice. This behavior can be explained by considering that, for scene 1, it is very frequent to have the same part of the scene visible in non-consecutive frames, since the camera is translating on a circular trajectory and not rotating. For scene 2, this is not the case because the non-consecutive frames having common parts are only the ones at the beginning and at the end of the sequence. Since the loop closure process relies on features detected in non-consecutive frames, the loop closure is more effective for scene 1 respect to scene 2. Indeed, in scene 2, for a high number of fragments the precision of the 3D reconstruction is very low.

For a lower number of frames (between 120 and 360 frames), for scene 1 (fig. 10), the trend is similar to the one in fig. 8, with a very small difference of SLAM performances by changing the number of fragments. This demonstrates that odometry and loop closure algorithms implemented in the reconstruction system can effectively find features among frames. Even if the number of frames is lower, odometry algorithms can be applied and consecutive frames, that are of few centimeters distant one from the other, still have parts of the framed scene in common. At the same time, the loop closure improves performances and makes the reconstruction more stable for scene 1, given a high number of features for non-consecutive frames.

For scene 2 (fig. 11), a number of fragments from 20 to 40 can reduce the ATE error and increase precision for a number of frames between 120 and 160, approximately. In this case, consecutive frames have less features in common and odometry can fail to reconstruct the 3D scene with a very low precision. Hence, increasing the number of fragments can reduce the point cloud alignment errors, by using the loop closure process, where the first and the last fragment have features in common and they can be used to close the trajectory loop. Although this final result may be not fully satisfactory (0.15-0.2 m of ATE and 0.4% of precision with a threshold τ=20mm), it can be used to minimize errors, in case only a low number of frames is available. For more than 160 frames, for a complete camera rotation trajectory, it is advisable to use a low number of fragments, if the number of features that are present in the scene allow to align the frames correctly with odometry.

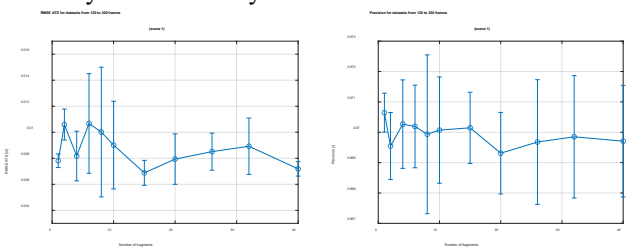


Fig. 10. RMSE ATE and precision for scene 1 for datasets with a number of frames between 120 and 360 respect to the number of fragments

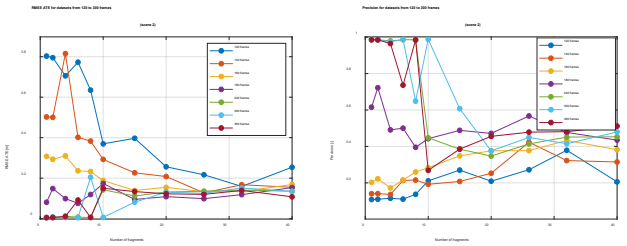


Fig. 11. RMSE ATE and precision for scene 2 for datasets with a number of frames between 120 and 360 respect to the number of fragments

6.3 Non-closing the trajectory loop

Loop closure is a process of recognizing previously visited scenes in a SLAM sequence to update the 3D and trajectory reconstruction process. This process extends also to frames where not all features are in common. However, for our trajectories, in which the last frame is coincident with the first, a closed trajectory could still be a great advantage [1], [28].

To study how the algorithm performances change if a trajectory loop is not closed, datasets covering 270° of the circular translation trajectory (scene 1) and of 270° of the rotation trajectory (scene 2) were set up. To make a comparison with results with complete trajectories (360° of circular translation or rotation), the number of frames was determined to maintain the same displacement or the rotation shift between consecutive frames. For example, for a complete trajectory with 360 frames, it was generated a non-closing loop trajectory of 270 frames; this means an angle rotation between consecutive frames of 1° for both cases. For the same objective, the number of fragments was selected to have the same portion of the trajectory covered by one fragment for both 360° and 270° trajectories. For instance, for a reconstruction with 20 fragments for 360° trajectory, the number of fragments of the corresponding 270° trajectory is 15. For this reason, fig. 12-13-14-15 show graphs with double X axis, one for 270° trajectories and the other for 360° trajectories. The only exception is for the case of 1 fragment in both 270° and 360° trajectories, in which the impact of loop closure is very low, given by the parameter preference loop closure odometry, as described in section 4.

Fig. 12-13 show the results of the previously explained test, for scenes 1-2 for datasets between 360 and 720 frames. The trend over the number of fragments for 270° trajectories is the same of 360° trajectories, but with lower quality results. For scene 1, for every number of fragments, the incomplete trajectory has lower performances, even for the case of 1 fragment. In this case, changing the number of fragments does not introduce any evident benefit, but if the trajectory loop is not completed, the performances are slightly lower. This behaviour is probably caused by the absence of last frames, on which loop closure process could be used inside fragments (this process is regulated by the parameter preference loop closure odometry).

In scene 2 (fig. 13), the RMSE ATE gap between complete and incomplete trajectories changes over the number of fragments. However, this trend is not observable for the precision of the 3D reconstruction. This can be explained by the much larger effect of changing the number of fragments and by the possibility to have few points of the reconstructed 3D scene, close to ground-truth, when the reconstruction has very low precision. As stated in [1], it should be noted that errors in the sensor trajectories reconstruction not always correspond to 3D map errors.

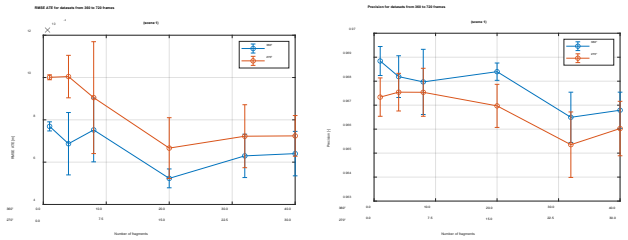


Fig. 12. RMSE ATE and precision for scene 1 for datasets with 360 to 720 frames to compare 270° and 360° circular translation trajectories

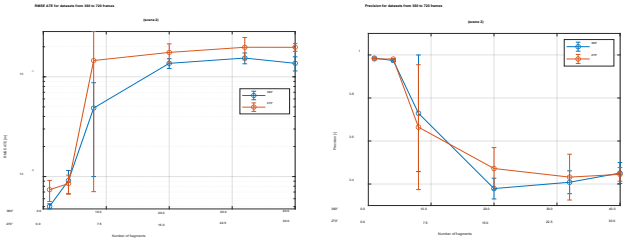


Fig. 13. RMSE ATE and precision for scene 2 for datasets with 360 to 720 frames to compare 270° and 360° rotation trajectories

In case of a lower number of frames, for the scene 1 (fig. 14), the difference between 270° and 360° trajectories is reduced, because of the lower difference of the number of frames between the two types of trajectories. Regarding the scene 2 (fig. 15), it is still evident that for a number of frames lower than 160 for complete trajectories using a high number of fragments improves the quality of the final result. In case of a number of frames of 120 and 90 for 360° and 270° trajectories respectively, it is visible an improvement for closing loop trajectory with respect to 270° trajectory. For other number of frames in fig. 15, it is not evident the performance enhancement for closing loop trajectories.

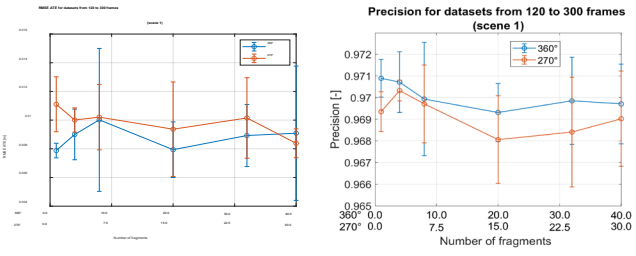


Fig. 14. RMSE ATE and precision for scene 1 for datasets with 120 to 360 frames to compare 270° and 360° rotation trajectories

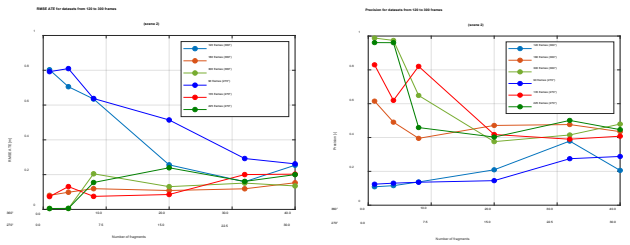


Fig. 15.RMSE ATE and precision for scene 2 for datasets with 120,180 and 300 frames to compare 270° and 360° rotation trajectories

7. Testing procedure on other datasets and SLAM or 3D reconstruction algorithms

As explained in section 1, this procedure can be applied to different algorithms and datasets, containing coloured point clouds. Examples of application of the proposed technique to different datasets and reconstruction algorithms is given here and in the following section. In this Section, with *scene 1* and *scene 2* we refer to the same scenes presented in Section 3, while scenes from 3 to 5 were obtained from the RGBDObject dataset [13], containing few objects (fig. 16). For these last 3 scenes, an orbital sensor trajectory around the 3D scene was chosen to create the sequence of images. Open3D reconstruction system and ElasticFusion SLAM algorithms were applied to the obtained image sequences to show how, with the proposed procedure, a comparison of different approaches can be easily done, both in fully controlled conditions and imposing any desired disturbance to simulate real environments. The trajectory is composed by 360 RGBD frames and the number of fragments of the Open3D reconstruction system was set to 4 with MDD equal to 70 mm, to include the majority of points within the depth range and to have enough fragments to take advantage of loop closure. ElasticFusion parameters were set starting from results from preliminary tests. These tests, joined with results shown in [29], evidenced that for testing scenes appropriate parameters can be ICP/RGB weight set to 10 out of 100 and a confidence threshold set to a low value of 4 to include a larger number of points for point clouds registering. The framerate used for the timestamp was set to 30, as typical 3D cameras acquisition rate. No disturbances were applied on datasets. The results are visible in table 1. In this case, the precision was computed with a threshold of 5mm, since the 3D scenes to reconstruct are smaller.

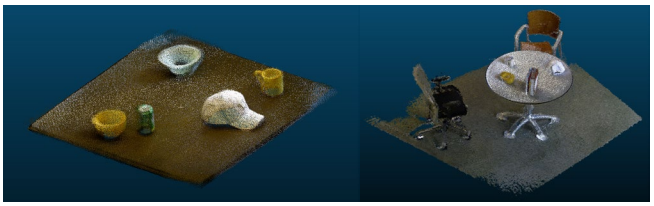


Fig. 16. The 3 additional scenes (scenes 3,4,5) [12]

Table 1. RMSE ATE and precision for scenes 1 to 5

	RMSE ATE [m]		Precision [-]	
	Open3D	Elastic Fusion	Open3D	Elastic Fusion
Scene 1	0.004	0.009	0.969	0.868
Scene 2	0.003	0.009	0.996	0.993
Scene 3	0.004	0.006	0.878	0.983
Scene 4	0.009	0.007	0.892	0.943
Scene 5	0.017	0.020	0.862	0.795

The results in table 1 are quite similar, between the two reconstruction systems. Values of RMSE ATE are very low and values of precision are high. This can be explained by the ability of the two systems to reconstruct a scene with quite small dimension and with a short sensor trajectory. More complex trajectories in wider environments can lead different results, as shown in [11] and in the following section. This example represents an application of the procedure presented in this article. The main advantage of the procedure is the ability to control each aspect of the simulation and the possibility to apply it to high accuracy colour point cloud dataset, e.g. a 3D point cloud acquired from a high accuracy scanner. In this sense, it is possible to study any aspect of a SLAM or 3D reconstruction algorithm, by isolating each specific aspect.

8. Application of a depth sensor noise model and of noise on trajectories

The application of this procedure of generating datasets is connected to the possibility of applying a noise model to simulate the acquisition of a specific sensor. An example of how an acquisition of a real sensor can be simulated is treated in this section.

To generate a sequence of RGBD frames with realistic disturbance on trajectory, the ground truth trajectories given in the dataset presented in [1] are considered. Coordinates and angles of these 11 trajectories were analysed separately and the FFT (Fast Fourier Transform) was applied to extract their frequency content. All these trajectories, truncated to have the same time history length, were used to compute the average of the frequency content. Frequency components below 2 Hz were characteristic of specific trajectories and, therefore, were deleted. Then, a time history of the disturbance signal due to

hand motion is generated with the IFFT (Inverse FFT). Different realization of the signals can be obtained applying every time a random phase. The time domain noise signal is bounded between +/- 1° for rotations and 0.08 m for translations.

For this example, the depth noise model was focused on modelling depth noise of Azure Kinect DK and Intel RealSense D415 sensors.

As stated in [30], the noise of Time-of-Flight sensors, as Azure Kinect, is inversely proportional to the amplitude of the Infrared signal received by the sensor, while in indoor conditions the disturbance of infrared daylight is limited. To model the depth error noise, a series of measures of a flat white wall at different distances, between 1 and 5 m, with a step of 0.5 m were performed. 100 frames were acquired for each distance and the standard deviation for each pixel at each distance was computed. The noise at each distance was modelled with a 3D Thin Plate spline, as proposed in [31]. These functions are visible in fig. 17. To determine the standard deviation at distances that are different from the ones acquired, a linear regression model was used to fit data for each pixel, since the noise is linearly influenced by the amplitude of the received infrared signal. In other terms, for each pixel p_{ij}, it is possible to consider the standard deviation σ_{ij} and a distance array d_{ij} such that:

$$\sigma_{ij} = a_{ij} d_{ij} + b_{ij} \quad (2)$$

where a_{ij} and b_{ij} are the coefficient for the linear regression at a specific pixel p_{ij}, obtained by fitting data at the distances of experimental tests. The mean and standard deviation of R^2 values of the fitting of each pixel are 0.983 and 0.011, respectively.

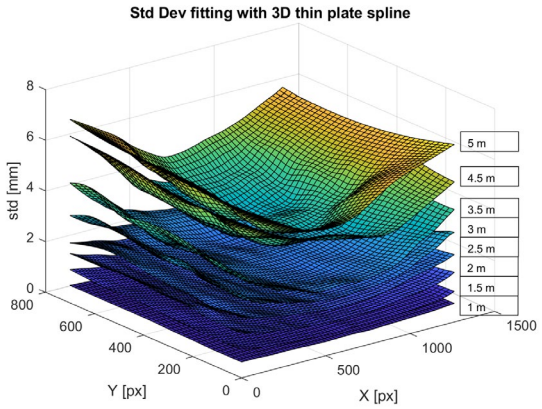


Fig. 17. The standard deviation noise model based on a 3D Thin Plate Spline at different distances

The resolution for this simulation is 1280x720 px and, to obtain realistic parameters, colour and depth are obtained by using intrinsic parameters extracted from the Kinect sensor, as described in the Open3D library documentation [23]. The Azure Kinect parameters are the following:

1. Focal length: fx=610.3 px, fy=610.3 px pixels;
2. Optical centre: cx=640 px, cy=360 px;
3. FOV: $F_x = 92.7°$, $F_y = 61.1°$.

Regarding Intel RealSense D415, the random error grows with respect to depth with a quadratic trend [23]. This trend, for a stereo camera, after calibration and image rectification, can be derived from the equation to compute depth:

$$z = \frac{Bf}{(x_R - x_L)} \quad (3)$$

Defining the disparity $d = x_L - x_R$, the depth error can be obtained by determining the depth derivative with respect to disparity. The error in depth δd is given by:

$$\delta z = \frac{z^2}{Bf} * \delta d \quad (4)$$

In this equation, the only unknown is δd and its value can be determined for each pixel by fitting the experimental random error maps in [22]. It is possible to generate a model based on fitting data for each pixel with equation (4). The mean and standard deviation of R^2 values of the fitting of each pixel are 0.944 and 0.058, respectively. The noise model for Intel RealSense D415 is visible in fig. 18.

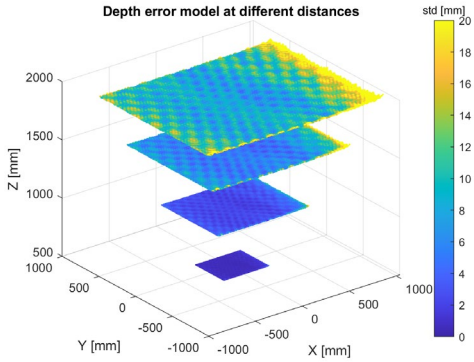


Fig. 18. The standard deviation noise model of Intel RealSense D415

For this test, a new trajectory and 3D scene was chosen [11]. The scene used in this case is shown in fig. 19. The trajectory used is a rectangular trajectory, where at each corner of the rectangle it is performed a rotation of 90° around the vertical axis. The noise to simulate the acquisition of a handheld scanner is added to the planned trajectory. The sensor noise is applied to simulate the acquisition with Azure Kinect DK and Intel RealSense D415 sensors, creating different datasets.

Fig. 19. 3D scene used for experiments to simulate acquisition with sensor noise and trajectory noise applied

In this case, for Azure Kinect simulation, the RGBD sequence is composed by 720 frames. The number of fragments for this test was set to 4, since it has been demonstrated in the section 6.3 that a low number of fragments is leads to a better result, when rotation is involved. MDD parameter was set to 100 mm, since the larger FOV of Azure Kinect can lead to a larger difference between corresponding points after rotation. For Intel RealSense D415, the simulation is composed by 3600 frames, since the device has a smaller FOV with respect to Azure Kinect. Moreover, as explained in section 6.3, a larger number of frames can reduce the error that is expected to be higher due to the higher level of random noise. The other parameters are the same of Azure Kinect ones. The results are shown in table 2.

Table 2. RMSE ATE and precision for the scene tested with Azure Kinect and Real

	RMSE ATE [m]		Precision [-]	
	Open3D	Elastic Fusion	Open3D	Elastic Fusion
Azure Kinect	0.018	0.153	0.962	0.443
RealSense D415	0.024	0.083	0.554	0.297

The results demonstrated that the larger level of random noise of RealSense D415 can significantly reduce precision of the resulting 3D map. Regarding the ATE, it is higher in case of ElasticFusion: this is likely due to the operating principle of ElasticFusion, which uses a real-time approach (i.e. adding one frame at a time to the full reconstruction, like in a real-time acquisition). The effect of the different random noise level of the two sensors does not show a significant influence on RMSE ATE. The ElasticFusion performances obtained are lower than the Open3D ones, as stated also in [11], however, ElasticFusion can be applied in real-time. With respect to [11], additional tests were performed by simulating the acquisition of Azure Kinect and comparing the performances with Intel RealSense D415.

From this example, it was shown how this procedure can be applied to test different sensors with different noise levels to determine their performances. The trajectory noise was applied as well, to consider realistic trajectories with respect to the one acquired by a handheld scanner.

9. Conclusions

In this article, we have presented a procedure to conduct a metrological evaluation of a 3D reconstruction algorithm with virtually created datasets, in order to control each specific condition or disturbance and its effect on the final result. These datasets were obtained from a point cloud acquired with a sensor with high accuracy (0.1 mm at 10 m distance), but the proposed procedure can be applied to high precision datasets and with any 3D reconstruction algorithm or SLAM algorithm in an offline mode. The metrological assessment was determined by calculating RMSE ATE and precision of 3D maps and trajectories, respectively. This procedure was applied to the Open3D reconstruction system.

The presented results show that, with a simulation environment, it is possible to study the response of the algorithm to disturbances, different acquisition conditions and processing parameters, by isolating each single effect. For the Open3D reconstruction system, it was determined the response to gaussian noise disturbances on colour and depth images. The results, in case of a sufficient number of frames acquired, e.g. 360 frames for selected trajectories, can reach high precision levels with typical RGBD sensors depth and colour noise levels. For example, with a depth disturbance lower than 30 mm of standard deviation, it is possible to obtain a RMSE ATE lower than 0.02 m and precision larger than 0.85.

Considering typical depth and colour noise disturbances, results evidenced the higher impact of depth noise with respect to colour noise. This is likely due to the more robust capability of detecting and matching features on RGB images, rather than on depth images. This can be considered as a condition for choosing a certain sensor for SLAM or 3D reconstruction.

The usage of the correct number of fragments can be crucial in case of a rotation trajectory to obtain precision larger than 0.9. In this case, the number of fragments to be used is lower than 6 for datasets with a rotation difference between 2 consecutive frames of 1°, while for a rotation difference higher than 2.25°, it is required to have a large number of fragments to reduce errors.

Not closing a 3D reconstruction trajectory can reduce the quality of the final result when the number of frames is larger than 360, while for a lower number of frames this effect is reduced or not clearly evident.

In the section 7, the application of the proposed method is extended to RGBDObject dataset and ElasticFusion SLAM

algorithm. Potentially, this method can be applied to any dataset composed by a coloured high accuracy point cloud and to any SLAM algorithm, which can be used offline, with data starting from files. During the creation of datasets with this procedure, it is also possible to simulate different sensors, by changing sensors parameters. Furthermore, as explained in section 8, sensors noise models were defined and applied, as well as the trajectory noise. These simulations can be useful to predict the metrological performances of different sensors, with different FOV and depth random noise. The models approach used for simulating Azure Kinect and RealSense D415 sensors can be applied also to other ToF and stereoscopic sensors.

Possible future research topics could be the extension of the method to other scenes and sensors error models can be refined to account for different environmental conditions, such as high reflectivity objects and outdoor applications.. Another improvement could be to use a mesh as input to the algorithm to permit the usage of lower density point clouds.

Acknowledgements

This work has been partially supported by the Italian Ministry of University and Research (MIUR) by the PRIN grant 2017HENS3L_003 (3DYNAMIC4.0).

References

[1] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, “A benchmark for the evaluation of RGB-D SLAM systems,” in IEEE/RSJ Int. Conf. Intell. Robots Syst., 2012, pp. 573–580.

[2] E. Yao, H. Zhang, H. Xu, H. Song, G. Zhang, “Robust RGB-D visual odometry based on edges and points”, Robotics and Autonomous Systems, Volume 107, September 2018, Pages 209-220.

[3] J. Stückler, S. Behnke, “Multi-resolution surfel maps for efficient dense 3D modeling and tracking”, Journal of Visual Communication and Image Representation, Volume 25, Issue 1, January 2014, Pages 137-147

[4] H. Dong, N. Figueroa, A. El Saddik, “Towards consistent reconstructions of indoor spaces based on 6D RGB-D odometry and KinectFusion”, IEEE International Conference on Intelligent Robots and Systems, 31 October 2014, Article number 6942798, Pages 1796-1803

[5] L. Nalpantidis, G. Sirakoulis, A. Gasteratos, “Non-probabilistic cellular automata-enhanced stereo vision simultaneous localization and mapping”, Measurement Science and Technology, Volume 22, Number 11, 14 October 2011

[6] M.Y.I Idris, H. Arof, E.M. Tamil, N.M. Noor, Z. Razak, “Review of Feature Detection Techniques for Simultaneous Localization and Mapping and System on Chip Approach”, Information Technology Journal, Volume 8, Issue 3, 2009, Pages 250-262

[7] F. Cao, Y. Zhuang, H. Zhang, W. Wang, “Robust Place Recognition and Loop Closing in Laser-Based SLAM for UGVs in Urban Environments”, IEEE Sensors Journal, Volume 18, Issue 10, May 2018, Pages 4242-4252

[8] S. A. Berrabah, H. Sahli, Y. Baudoin, “Visual-based simultaneous localization and mapping and global positioning system correction for geo-localization of a mobile robot”, Measurement Science and Technology, Volume 22, Number 12, 15 November 2011

[9] H. Weijian, W. Kaiwei, C. Hao, C. Ruiqi and Y. Kailun, “An indoor positioning framework based on panoramic visual odometry for visually impaired people”, 2020 Meas. Sci. Technol, , Volume 31, Number 1

[10] Y. K. Yu, K. H. Wong, S. H. Or, M. Y. Chang, “Robust 3-D motion tracking from stereo images: A model-less method”, IEEE Transactions on Instrumentation and Measurement., vol. 57, no. 3, pp. 622–630, Mar. 2008

[11] J. Park, Q.Y. Zhou, V. Koltun, “Colored Point Cloud Registration Revisited”, Proceedings of the IEEE International Conference on Computer Vision, Volume 2017-October, 22 December 2017

[12] S. Choi, Q.-Y. Zhou, and V. Koltun, “Robust Reconstruction of Indoor Scenes”, Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Volume 07-12-June-2015, October 2015, Pages 5556-5565

[13] K. Lai, L. Bo, X. Ren, D. Fox, “A Large-Scale Hierarchical Multi-View RGB-D Object Dataset”, In IEEE International Conference on Robotics and Automation (ICRA), May 2011.

[14] T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, S. Leutenegger, “ElasticFusion: Real-Time Dense SLAM and Light Source Estimation”, The International Journal of Robotics Research, 2016.

[15] S. Shah, D. Dey, C. Lovett, A. Kapoor, “AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles”, Field and Service Robotics, 2017

[16] N. Koenig, A. Howard, “Design and use paradigms for Gazebo, an open-source multi-robot simulator”, 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) Volume 3, Pages 2149-2154, 200

[17] A. Handa, T. Whelan, J.B. McDonald, A.J. Davison, “A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM” Proceedings of the IEEE International Conference on Robotics & Automation, 1524-1531. Hong Kong, 2014.

[18] P. Cignoni, C. Rocchini, R. Scopigno, “Metro: Measuring Error on Simplified Surfaces”, Computer Graphics Forum, Volume 17, Issue 2, June 1998, Pages 167-174

[19] “Cloud-to-Cloud Distance cloudcompare.” https://www.cloudcompare.org/doc/wiki/index.php?title=Cloud-to-Cloud_Distance

[20] C. Dinesh, G. Cheung, I. V. Bajic, C. Yang, “Fast 3D Point Cloud Denoising via Bipartite Graph Approximation & Total Variation”, 2018 IEEE 20th International Workshop on Multimedia Signal Processing (MMSP)

[21] R.B. Rusu, Z. C. Marton, N. Blodow, M. Dolha, and M. Beetz. “Towards 3D Point Cloud Based Object Maps for Household Environments”. Robotics and Autonomous Systems Journal. 2008.

[22] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin and C. T. Silva, "Computing and Rendering Point Set Surfaces", IEEE Transactions on Visualization and Computer Graphics, Volume 9, Issue 1, January 2003, Pages 3-15

[23] S. Giancola, M. Valenti, R. Sala, “A Survey on 3D Cameras: Metrological Comparison of Time-of-Flight, Structured-Light and Active Stereoscopy Technologies”, 2018, in SpringerBriefs

[24] Guide to the Expression of Uncertainty in Measurement (GUM), "Evaluation of measurement data – Guide to the expression of uncertainty in measurement", JCGM 100:2008 (GUM 1995 with minor corrections)
[25] "Open3D reconstruction system documentation": http://www.open3d.org/docs/release/tutorial/ReconstructionSystem/system_overview.html
[26] B. Curless and M. Levoy. "A volumetric method for building complex models from range images". Proceedings of the ACM SIGGRAPH Conference on Computer Graphics, 1996, Pages 303-312
[27] B. Horn, "Closed-form solution of absolute orientation using unit quaternions," Journal of the Optical Society of America A, vol. 4, pp. 629–642, 1987;
[28] R. Munguía, A. Grau, "Closing loops with a virtual sensor based on monocular SLAM", IEEE Transactions on Instrumentation and Measurement, Volume 58, Issue 8, 2009, Pages 2377-2384
[29] "Design-Space Exploration for Dense 3D Scene Understanding: Exploring ElasticFusion", Imperial College of Science, Technology and Medicine, MSc. Thesis
[30] T. Edeler, K. Ohliger, S. Hussmann, A. Mertins: "Time-of-flight depth image denoising using prior noise information". Circuit Serial Programming, 2010
[31] A. Belhedi, A. Bartoli, S. Bourgeois, K. Hamrouni, P. Sayd, V. Gay-Bellile, "Noise modelling and uncertainty propagation for TOF sensors", Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Volume 7585 LNCS, Issue PART 3, 2012, Pages 476-485

Daniele Marchisotti is a PhD student at the department of mechanical engineering of Politecnico di Milano. He received the MSc degree in Mechanical Engineering. His research interests involve vision-based measurement with RGBD sensors in dynamic environments, mainly focused on the area of the application and improvement of SLAM algorithms and vibration measurement.

Emanuele Zappa (M'10) is Full Professor in Mechanical and Thermal Measurements at Politecnico di Milano, Italy. He is Associate Editor of the IEEE Transaction on Instrumentation and Measurement. The scientific activity is mainly in the field of mechanical and image-based measurements.