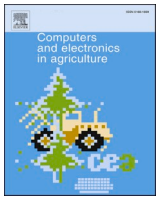
Computers and Electronics in Agriculture 180 (2021) 105929

Contents lists available at ScienceDirect

Computers and Electronics in Agriculture

journal homepage: www.elsevier.com/locate/compag



Original papers

Fast and accurate GPU-accelerated, high-resolution 3D registration for the robotic 3D reconstruction of compliant food objects

Ulrich Johan Isachsen [a,*], Theoharis Theoharis [a], Ekrem Misimi [b]

[a] NTNU, Department of Computer Science, Sem Sælandsvei 9, Trondheim, Norway
[b] SINTEF Ocean, Brattørkaia 17C, 7010 Trondheim, Norway

ARTICLE INFO

Keywords:
Point cloud registration
RGB-D sensor
Food production
Graphics Processing Unit (GPU) registration
Automation

ABSTRACT

If we are to develop robust robot-based automation in primary production and processing in the agriculture and ocean space sectors, we have to develop solid vision-based perception for the robots. Accurate vision-based perception requires fast 3D reconstruction of the object in order to extract the geometrical features necessary for robotic manipulation. To this end, we present an accurate, real-time and high-resolution ICP-based 3D registration algorithm for eye-in-hand configuration using an RBG-D camera. Our 3D reconstruction, via an efficient GPU implementation, is up to 33 times faster than a similar CPU implementation, and up to eight times faster than a similar library implementation, resulting in point clouds of 1 mm resolution. The comparison of our 3D reconstruction with other ICP-based baselines, through trajectories from 3D registration and reference trajectories for an eye-in-hand configuration, shows that the point-to-plane linear least squares optimizer gives the best results, both in terms of precision and performance. Our method is validated for the eye-in-hand robotic scanning and 3D reconstruction of some representative examples of food items and produce of agricultural and marine origin.

1. Introduction

Robotic manipulation of produce in primary production and processing in agriculture, ocean space and food industries is a growing demand in order to increase the production efficiency and competitiveness but also, through higher degree of automation, to be able to better tackle crisis and lockdown situations, when activity of human operators in the field and in the processing plants may be restricted. The majority of complex manipulation operations in these sectors are still manual based due to the lack of reliable robot-based solutions. Since typical produce from these sectors such as fruits, vegetable, meat or fish are compliant objects, a significant hinder for a higher degree of automation is the challenge that robotic manipulation of such objects incurs, but also compelling scientific and technological challenges such as those linked to the visual-based perception and manipulation. Obtaining good 3D image data is a key prerequisite in vision-based robotic manipulation applications (Calli and Dollar, 2016; Cherubini et al., 2018; Misimi et al., 2018) and visual servoing (Agravante and Chaumette, 2017; Pedersen et al., 2020). Consumer 3D scanner technology has made cheap, lightweight and reliable RGB-D cameras available for vision-based robotic manipulation tasks. This has opened up for many novel robotic applications with an increased ability of the robots to handle various complex objects with unknown shapes such as 3D compliant food objects (Misimi et al., 2018) and clothing (Yuan et al., 2018). To manipulate such objects in a real-time manner, there is a need for an efficient, high resolution 3D reconstruction algorithm and implementation. Given a set of partially overlapping 3D scans, registration is the process of aligning these scans such that the distance between overlapping parts is minimized. The iterative closest point (ICP) algorithm and its variations, first proposed by Besl and McKay (Besl and McKay, 1992), are widely used and can serve as a starting point for such a task. However, given the speed requirements necessary for making scanning decisions while the robot arm is moving in a manipulation task (Misimi et al., 2018; Misimi et al., 2016), GPU implementations are necessary to enable real-time 3D registration and enhance the robot's manipulation efficiency. To this end, we present an extensive evaluation and a GPU implementation of the ICP algorithm variants. The setup is a collaborative/industrial robot arm with an eye-in-hand RGB-D sensor and the goal is to present a multi-view combined model of an unknown object, that can be used in a variety of vision-based robotic manipulation tasks, such as grasping. The

* Corresponding author.
E-mail addresses: ulrich@isachsen.org (U.J. Isachsen), theotheo@ntnu.no (T. Theoharis), Ekrem.Misimi@sintef.no (E. Misimi).

https://doi.org/10.1016/j.compag.2020.105929
Received 13 September 2020; Received in revised form 27 October 2020; Accepted 25 November 2020
Available online 23 December 2020

operating environment is not constrained, but is assumed to be clean and sterile.

In this work, we present a real-time 3D reconstruction approach applicable to robotic scanning and perception with a RGB-D sensor in the context of robotic manipulation of 3D compliant objects of agricultural and marine origin. The paper makes two main contributions to 3D registration. The first is a GPU implementation of a variation of the ICP algorithm that makes use of a Linear Least Squares (LLS) optimizer to compute the transformation in the ICP variants. We compare this implementation to other variations of the ICP algorithm by tracking and comparing the generated trajectory by the ICP variants against a ground truth trajectory generated by the robot arm, where the scanner is mounted on the end-effector. The second contribution is an evaluation of different variations of the ICP algorithm in the context of the 3D registration of objects, coming from food, agricultural and ocean space domains, placed on a semi-planar surface, where an RGB-D sensor is mounted on a robot arm in an eye-in-hand configuration. The approach we propose is close to that of Lehnert et al. (2016), but the novelty is that the 3D registration differs in that we use frame to model registration, which is real-time enabled using point cloud simplification after each captured frame. In addition, we chose not to depend on a voxelization method due to the potential loss of precision as a result of sub-sampling that results from the voxelization, which is the case in the Kinect Fusion algorithm (Newcombe et al., 2011). Moreover, we focus on a real-time robotic application with eye-in-hand configuration for registration of 3D compliant objects coming from agriculture and ocean space such as fruits and vegetables but also muscle food products such as meat and fish fillet portions.

The accumulated models from our proposed algorithm are shown in Figs. 1–3. We evaluate the accuracy of our results using the Absolute Trajectory Error (ATE) (Sturm et al., 2012), by comparing the transformations obtained from the registration algorithm to the true transformation of the robot arm (See Section 4.3). Fig. 4 shows the estimated positions of the RGB-D sensor generated by the proposed algorithm together with the position of the robot's end-effector while scanning the objects of Fig. 1.

The paper is organized as follows. After the Related Work, in the Methodology section we present a GPU implementation of the point-to-plane linear least squares ICP algorithm. In Results and Discussion, the proposed method is shown to achieve significant speed-up compared to an equivalent CPU implementation, without noticeable loss of precision.

2. Related work

2.1. ICP variants

Point cloud registration has been an active field of study since the introduction of the first Iterative Closest Point (ICP) formulation by Besl and McKay (1992). Their method was based on a point-to-point quaternion optimizer. Simultaneously, Chen and Medioni (1991) proposed a point-to-plane optimization strategy which was shown by Rusinkiewicz and Levoy (2001) to give faster convergence. Our work is based on these two optimizers where we evaluate them in the context of a real-time framework in a sterile environment for the 3D registration of solid compliant food and similar objects.

2.2. GPU-based registration

Qiu et al. (2009) showed that a GPU can be employed to accelerate the ICP algorithm. Their method was based on constructing a k-dimensional (k-d) tree on the CPU and performing the nearest neighbour search on the GPU. The GPU search is made possible by fixing the size of the search queue, which will approximate the search. This method gave a speedup factor of up to 88 compared to a similar CPU implementation, where a point-to-point error metric (see Besl and McKay, 1992) was used to evaluate the convergence of the algorithm for different queue sizes. The implementation was a standard ICP with a Singular Value Decomposition (SVD) optimizer. A natural extension to this is to perform k-d tree construction on the GPU. However, Hu et al. (2015) showed that the speedup potential of such an approach is lower than the one based on the GPU correspondence search.

For real-time registration, Kinect Fusion is a popular choice and is based on voxelization to obtain faster registrations (Newcombe et al., 2011). Lehnert et al. (2016) in a setup similar to ours in terms of sensor setup and robot mounting, found that the Kinect Fusion algorithm was suited to food object reconstruction using frame to model tracking. They compared it to simple ICP and normal distributions transform (NDT) variations using frame to frame tracking which resulted in drift over time. In comparison to the ICP and NDT variations of Lehnert et al. (2016), our work differs in the fact that we use ICP for frame to model, which is real-time enabled using point cloud simplification after each captured frame. In contrast to Lehnert et al. (2016), we chose not to use voxelization method like presented in Kinect Fusion due to the potential loss of precision as a result of sub-sampling from the voxelization.

2.3. Registration speed-up using point cloud simplification methods

Other space simplifications such as edge extraction can also be an alternative to speed up registration. Examples include Choi et al. (2013) who used an edge-detection strategy and evaluated the precision of the registration using Relative Pose Error (RPE) metrics (see Sturm et al., 2012). Edge extraction can greatly improve registration time (and possibly quality), but can reduce the generality of the algorithm. However, it will also depend on having certain features available. In a robot-based 3D scanning application, the environment is expected to be sterile, which is the reason we don't want to depend on feature based registration.

2.4. Other registration methods

Chen et al. (2016) evaluated a registration algorithm with an eye-in-hand setup, quite similar to our setup. They show promising results in terms of registration quality when using precise measurement equipment but it is not concerned with real-time requirements, and is not a standard ICP algorithm, which can be found in many popular point cloud based libraries.[1]

Our work focuses on the ICP algorithm for local registration, which is based on the assumption that the movement between frames is limited, making global optimization strategies (such as Zhou et al., 2016) unnecessarily complicated for the task. The field of registration techniques is extremely large. Methods from other application areas, such as the medical area, have been considered, such as the method by Pratikakis et al. (2003) for the registration of deformable medical objects. However, they have not been deemed relevant due to their assumption of a dense representation of the data. In this work, we specifically address the problem of real-time 3D registration of 3D compliant food objects of agricultural and marine origin, in the context of robotic scanning and manipulation, using a RGB-D camera that produces high resolution, without assuming any structure of the point-clouds.

3. Methodology

Each successively captured RGB-D image is converted to a colored point cloud S. Note that color is not used as part of the registration algorithms but is used for visualization purposes (this is because many depth scanners don't support color). An outline of the algorithm is shown in Algorithm 1.

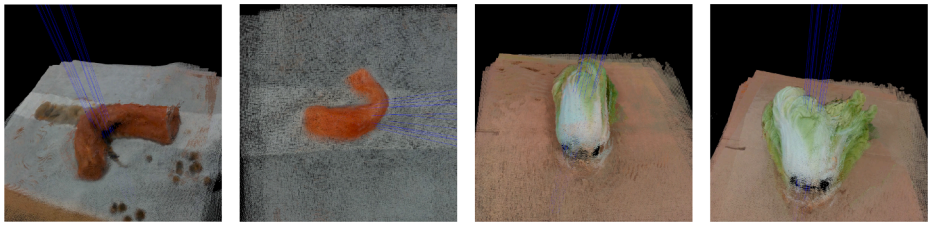
[1] Refer to Holz et al. (2015) on how to use a point cloud library for registration, including strategies for initial alignment and reducing point cloud size to speed up registration.

(a) Apple (b) Banana (c) Fruits (d) Pear

Fig. 1. Accumulated point clouds of example fruits by the proposed algorithm using the point-to-plane linear least squares optimizer and nearest point correspondence estimation, from scanning with an Intel RealSense SR300 mounted on the end effector of a robot arm. The parameters are given in Section 3. The shown 3D registrations are a result of the accumulation of 35 point clouds, with an average processing of 2.23 s per image using the GPU implementation outlined in Section 3.

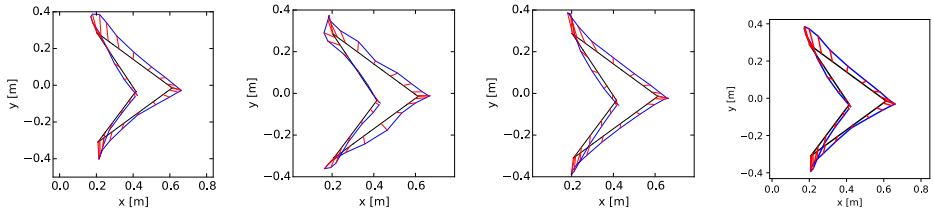
(a) Lettuce (b) Pork chop (c) Pork filet (d) Salmon

Fig. 2. A similar method is used as in Fig. 1, but the images was captured with a handheld Intel RealSense D435 RBG-D sensor. In this test set, there were significant amounts of noise in the environment besides the table itself. In this setting, our point correspondence rejection strategy (Section 3.4) was employed in order to get good registrations. The registration parameters are given in Section 3. Reference objects were added to the scene to give the registration algorithm some features other than the flat objects themselves. The Figure shows that the suggested framework is able to register free-hand scans and that relatively high-quality scans can be obtained from consumer grade scanners. The captures are accumulations of: (a) 20 images, (b) 14 images, (c) 30 images and (d) 22 images.



(a) Pork loin (b) Salmon (c) Lettuce (1) (d) Lettuce (2)

Fig. 3. Accumulated point clouds of the pork loin, salmon fillet, and lettuce after performing a real-time 3D registration, based on our method, resulting from motion of the RGB-D sensor, mounted on a robot arm, along the motion scan trajectories. All point clouds are at 1 mm resolution. (a), (c), and (d) Are registrations from 15 point clouds, while (b) are registrations from 18 point clouds.



(a) Apple (b) Banana (c) Fruits (d) Pear

Fig. 4. Estimated trajectory of the registrations shown in Fig. 1 (blue line), plotted against the reference trajectory obtained from the robot arm (black line). The estimated trajectories are relatively close to the (ground truth) reference trajectories where the error is shown in red. This is expected from the good quality of the resulting point clouds of Fig. 1. Note that most of the figures have a near-perfect loop closure, which also indicates a good registration. The camera was approximately 40 cm above the ground plane during the scans. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Algorithm 1. GPU based registration

(continued on next column)

(continued)

(continued on next page)

(*continued*)

input : Destination point cloud D with normals
input : Source point cloud S
input : Initial transformation $\hat{\mathbf{M}}$ (Previous registration)
output: Transformation matrix $\hat{\mathbf{M}}_{\text{final}}$

CPU: $\hat{\mathbf{M}}_{\text{final}} \leftarrow \hat{\mathbf{M}}$
CPU: $D_{\text{tree}} \leftarrow$ Construct left balanced k-d tree from D
CPU: Transfer D_{tree}, S to the GPU
while *convergence criteria not reached* **do**
  CPU: Transfer $\hat{\mathbf{M}}$ to the GPU
  GPU: $S \leftarrow \hat{\mathbf{M}}S$
  GPU: $D_C \leftarrow$ Correspondences(D_{tree}, S)
  GPU: $D_C \leftarrow$ CorrespondenceRejection (D_C)
  GPU: $\mathbf{A}^T\mathbf{A} \leftarrow$ Compute from D_C using summing-tree
  GPU: $\mathbf{A}^T\mathbf{b} \leftarrow$ Compute from D_C using summing-tree
  GPU: transfer $\mathbf{A}^T\mathbf{A}$ and $\mathbf{A}^T\mathbf{b}$ to the CPU
  CPU: $\mathbf{x}_{\text{opt}} = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{b}$
  CPU: $\hat{\mathbf{M}} \leftarrow$ MatrixFromParameters$(\mathbf{x}_{\text{opt}})$ (See Low [2004])
  CPU: $\hat{\mathbf{M}}_{\text{final}} \leftarrow \hat{\mathbf{M}}\hat{\mathbf{M}}_{\text{final}}$
end

3.1. Correspondence estimation, k-d tree construction and search

The main part of our method and algorithm implementation is build upon the work by Qiu et al. (2009). A k-d tree was constructed for the points in the accumulated model on the CPU. The k-d tree is left-balanced, giving it a dense, GPU friendly, representation. It was then transferred to the GPU where the nearest neighbour search is done in parallel using a fixed-size stack. In our work, the stack size was set to 20. The next step in the ICP algorithm was to find a transformation that minimizes the distance of the found correspondences. A correspondence set D_C consists of triplet elements $D_{C,i} = (s_i, d_i, n_i)$, where s_i is from source cloud S, their correspondences d_i from destination cloud D with n_i being the normal of d_i for $i = 1 \ldots N$ where N is the number of points in S. The source cloud is the point cloud that is to be transformed to be aligned with the previously registered point clouds accumulated into destination cloud D.

3.2. Transformation estimation

Fitzgibbon (2002) introduced the Levenberg-Marquardt (LM) optimizer for registration. Low (2004) extended this work and showed how the LM problem can be expressed as a Linear Least Squares (LLS) problem by assuming that the rotation is small. This is a fair assumption in our setting where the registration is performed continuously from a camera mounted on a robot arm. Building on Low's method and using a similar notation, we present an extension for the optimization, which is more suitable for a GPU implementation.

The goal is to find a transformation matrix $\widehat{\mathbf{M}}_{\text{opt}}$ such that the point to plane error is minimized: $\widehat{\mathbf{M}}_{\text{opt}} = \arg\_\min_{\widehat{\mathbf{M}}} \sum_i \left(\left(\widehat{\mathbf{M}} \cdot s_i - d_i \right) n_i \right)^2$.

$\widehat{\mathbf{M}} = \mathbf{T}(t_x, t_y, t_z)\widehat{\mathbf{R}}(\alpha, \beta, \gamma)$ is a rigid transform composed of the translation $\mathbf{T}$ and rotation $\widehat{\mathbf{R}}$, and can be expressed by the parameter vector $\mathbf{x} = \begin{pmatrix} \alpha & \beta & \gamma & t_x & t_y & t_z \end{pmatrix}^T$, where α, β and γ are the Euler-angles for the rotation $\widehat{\mathbf{R}}$ and t_x, t_y and t_z are the translational components in $\mathbf{T}$.

By assuming that the rotation $\widehat{\mathbf{R}}$ is small, the matrix can be simplified such that the problem can be approximated as a linear least-squares (LLS) problem. For each value of i, the expression $\left(\widehat{\mathbf{M}} \cdot s_i - d_i \right) n_i$ is inserted as a row into $\mathbf{Ax} - \mathbf{b}$, where $\mathbf{A}$ is a matrix of size $N \times 6$ and $\mathbf{b}$ is a column vector of length N for 6 variables and N points (see Low (2004) for details on the values that go into $\mathbf{A}, \mathbf{b}$ and the 6 variables that make up $\mathbf{x}$).

Using the $\mathbf{Ax} - \mathbf{b}$ formulation, the optimal transformation is solved by a linear least squares formulation $\mathbf{x}_{\text{opt}} = \arg\_\min_{\mathbf{x}} |\mathbf{Ax} - \mathbf{b}|^2$, which Low (2004) showed can be solved by $\mathbf{x}_{\text{opt}} = \mathbf{A}^+\mathbf{b}$. As the columns of $\mathbf{A}$ are linearly independent and only have real valued entries, $\mathbf{A}^+$ can be rewritten as $\mathbf{A}^+ = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T$. This can be assumed true because A consists of 6 columns and N rows, where each row come from separate correspondence pairs.

Now, the expression $\mathbf{x}_{\text{opt}} = \mathbf{A}^+\mathbf{b}$ can be rewritten as $\mathbf{x}_{\text{opt}} = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\left(\mathbf{A}^T\mathbf{b}\right)$. The result of this is that the multiplications in the entries of the 6x6 matrix $\mathbf{A}^T\mathbf{A}$ and 6x1 vector $\mathbf{A}^T\mathbf{b}$ only have dependencies from the same index i, and can be computed from a summing tree on the GPU from the individual correspondences i. More specifically, we compute matrix $(\mathbf{A}^T\mathbf{A})_i$ and vector $(\mathbf{A}^T\mathbf{b})_i$ for each correspondence pair i (using s_i, d_i and n_i). Then, matrix $\mathbf{A}^T\mathbf{A}$ and vector $\mathbf{A}^T\mathbf{b}$ are computed using a summing tree for all correspondence pairs $(\mathbf{A}^T\mathbf{A})_{1..N}$ and $(\mathbf{A}^T\mathbf{b})_{1..N}$. See Algorithm 1 for an overview of the GPU implementation.

The transformation $\widehat{\mathbf{M}}_{\text{opt}}$ is found from $\mathbf{x}_{\text{opt}}$ (see Low, 2004). All transformation estimators are given as the transformation of the previous aligned scan as an initial estimate.

3.3. Convergence criteria

For this work, the transformation epsilon was set to 10^{-8}, the fitness epsilon to 10^{-8} and the max iteration count was 50. These were three possible convergence criteria considered and also available in point cloud library.[2]

3.4. Point correspondence rejection

A point correspondence rejection strategy was implemented, similar to the boundary rejection strategy proposed by Rusinkiewicz and Levoy (2001). The rejector was specified as a box with user-specified boundaries. All correspondences that are outside the box are rejected. As the ICP algorithm iterates, the box is transformed using the transformation applied to the point cloud. The advantage of the box rejector is that no data structure is required in the point-cloud representation.

This strategy was applied to the scans for Figs. 1b, and the free-hand scans for 2a, b, c, and d.

The following $(x_{\min}, x_{\max}, y_{\min}, y_{\max}, z_{\min}, z_{\max})$ boundaries (measured in meters) were used:

Banana	(−0.1,	0.1,	−0.05,	0.05,	0.0001,	10)
Free-hand	(−0.20,	0.20,	−0.20,	0.20,	0.0001,	10)

The coordinates are given in camera space, where the x-axis is horizontal, the y-axis is vertical and the z-axis is towards the object.

3.5. Normal computations

The point-to-plane linear least squares transformation method

[2] See Holz (Holz et al., 2015) for more details about convergence criteria in PCL.

described by Low (2004) requires that normals are computed. This is done using Point Cloud Library (PCL), as described by Rusu (2009). The search radius was set to 5 mm for all test sets[3].

3.6. Voxel grid filter

Downsampling was performed on the accumulated model point cloud after merging with the captured point cloud. This was done using a voxel-grid filter implemented in PCL, using the method described by Rusu (2009). This step is required so that the number of points in the accumulated model stays stable and constant, keeping a stable performance independent of the number of registered captures. The grid size is set to 1 mm, which is about the same resolution as that of the 3D scanner at the scanning range used.

3.7. Additional ICP variations

The additional ICP variations plotted in Figs. 5, 6, 7, and 9, some of which are used as baselines for comparison to our method, are based on the registration framework in PCL.

4. Evaluation

4.1. Physical setup

The algorithms were run on a computer with an AMD ryzen 1700X processor, and an NVidia GTX 1080Ti GPU. An Intel RealSense SR300 camera was mounted on the end effector of a Franka Panda robot arm.

4.2. Test sets

The test sets are based on different scan trajectories (Section 4.2.1) and objects (Section 4.2.2).

4.2.1. Scanning motion trajectories

The robotic scan trajectories shown in Fig. 4 and the free-hand scans of Fig. 2 will be referred to throughout the evaluation section of the paper. For the robotic scanning with a RGB-D sensor, a pre-defined deterministic scanning motion was used to generate a motion trajectory that enabled the RGB-D sensor to scan all sides of the object. Other motion trajectories were also trialed, and gave similar results. During all robot-based scanning, the object was centered in the camera view, which simplified the initial alignment for the ICP algorithm, in the sense that using the previously found transformation as initial alignment is sufficient. The motion of the end-effector of the robot arm was a linear acceleration between waypoints with an upper limit of 0.05 rad/s^2 for rotation and 0.1 m/s^2 for translation, giving the trajectories shown in Fig. 4. Captures are made with a frame-rate of 10 Hz, but fewer images are used because the registration generally takes longer than 100 ms. The internal pose estimation of the robot was done at 1000 Hz (the control-loop rate of the robot arm).

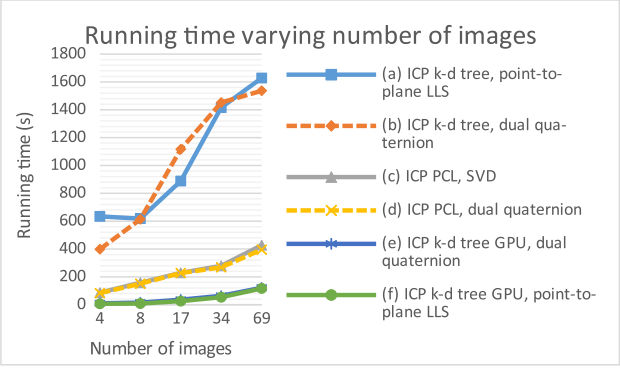


Fig. 5. Total running time of different variants of the ICP algorithm when varying number of point clouds to register (frame rate) on the fruits dataset. Our method (f) and the other GPU implementation (e) are the fastest ones, followed by PCL implementations.

[3] Normal computation takes about 0.7 s for a scan image when done on the CPU and 0.05 s when done on the GPU. There are many options for performing this computation and it has not been included in the timings.

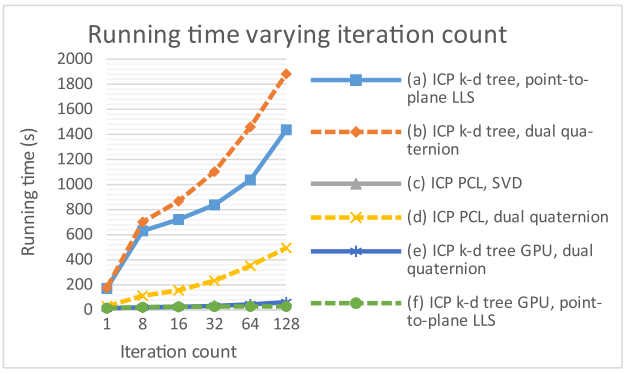


Fig. 6. The running time of different variants of the ICP algorithm on 18 depth images when varying the number of iterations on the fruits dataset. Not all iterations are necessarily run as the termination criteria may first be met. This is observed in the point-to-plane LLS algorithms, which converges to the correct transformation after only a few iterations. The figure shows that the running times of the GPU implementations are significantly lower than the CPU based versions.

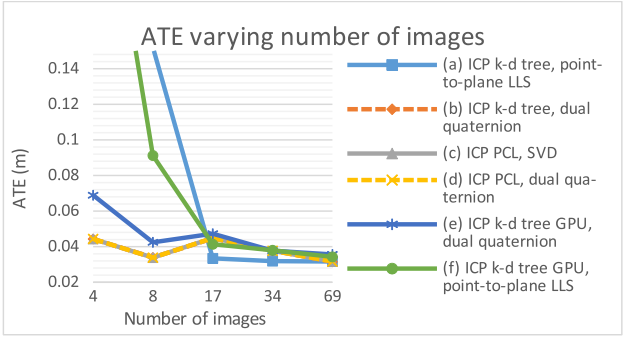


Fig. 7. The absolute trajectory error (ATE) for the registration of the fruits dataset, varying the number of images. The figure shows that there is no noticeable difference between the registration methods, except for the point-to-plane LLS transformation estimators where large errors are noted if the number of images drops below 10. Line (b) and (c) is hidden behind (d).

4.2.2. Food objects in evaluation set

The method was evaluated by scanning different food objects whose scans are shown in Figs. 1 and 2. These particular food items were selected as representative examples and due to their difference in size, shape, texture, and compliancy properties, and as representatives of some of the most common fruits/vegetables and muscle foods (meat and fish) consumed in national and global scale. These objects have all different texture, mechanical and optical properties and different types of features. From the perspective of the robot-based visual perception, small-sized produce with circular shape, such as the apples, are considered challenging objects to scan because of their geometrical rotational symmetry and the small size. As representatives of muscle foods from agriculture and ocean space, compliant produce such as meat

- pork loin, and fish - salmon fillet portions were used for the evaluation of the robot-based real-time 3D reconstruction. Robotic 3D reconstruction of these food items can be seen in Fig. 3.

4.3. Evaluation metrics

The *estimated trajectory* is found by applying the transformations found by the proposed registration algorithm to the camera start position, thus tracking the frame-to-frame movement. The *reference trajectory* (ground truth) camera movement is estimated from the end effector of the robot arm. A top-down view of these trajectories can be seen in Fig. 4. A quantitative measure of the registration precision is shown in Figs. 7 and 9 in terms of the absolute trajectory error (ATE) as described by Sturm et al. (2012).

4.4. Registration results

Figs. 1 and 2 present the accumulated point clouds after running the GPU accelerated ICP algorithm with the point-to-plane LLS transformation estimator. These figures show that the proposed algorithm produces good quality 3D reconstructions of compliant objects from scanning with a RGB-D sensor mounted on a robot arm, and the visual quality of the point clouds are similar to e.g. Kinect fusion (Newcombe et al., 2011) and Morell-Gimenez et al. (2014). The figures also show that consumer RGB-D based scanners give satisfactory reconstructions of the objects.

4.5. Trajectory estimation

Fig. 4 shows that the estimated trajectories from the registration are close to the reference trajectories, meaning that the registration algorithm is correctly aligning each capture from the sensor. Some measurement errors from the RGB-D sensor and robot arm pose estimation are still present. Similar to the reference trajectory, the estimated trajectory ends close to the starting point, indicating that the registration has little drift from frame to frame (such a drift is demonstrated in Fig. 8). Choi et al. (2013) shows a similar trajectory comparison with a similar error. Newcombe et al. (2011) also shows trajectory estimations, though their trajectories are not compared to any reference trajectory.

4.6. Performance

From Fig. 5 it is possible to read that the GPU implementations have a speed-up factor of 14 times that of a similar CPU implementation and 4.5 times that of a PCL implementation. The fastest variant requires about 1.5 s per image, which can be considered real-time for a robotic scanning application where the movement of the robot arm typically requires some time (especially in a collaborative setting, where it might work together with humans).

Reading Fig. 6, the best speedups appear when comparing algorithms on the CPU vs the GPU; in the case of the dual quaternion estimator a speedup of 30 is observed, while for the point-to-plane LLS method the speedup is 48. The dual quaternion GPU method is 8 times faster than the library PCL implementation. This speed-up is comparable to that of Qiu et al. (2009) who obtained a speed-up factor of 88 using a queue size of 1, while in our work, we used a stack size of 20. The iteration count also shows that the point-to-plane LLS transformation estimator requires fewer iterations to give a good convergence compared to other ICP methods. Fig. 9 shows that 10 iterations of a point-to-plane LLS transformation estimator is comparable to 100 iterations of a point-to-point dual quaternion transformation estimator.

Both Newcombe et al. (2011) and Choi et al. (2013) show slightly faster registration per frame than our work, but this is expected given that our work focuses on a robotic scanning applications, where a high resolution point cloud is a higher priority than registration speed. Kinect fusion (Newcombe et al., 2011) depends on voxelization and resolution must be compromised for larger scenes, and Choi et al. (2013), was concerned with tracking the 3D scanner and not the visual quality of an accumulated model. Since the idea behind our work is to be able to generate and use the 3D object models for robotic manipulation tasks, high resolution point clouds are prerequisite for prehensile grasping or other manipulations tasks which involve interactions with the 3D compliant food object of interest to be manipulated. Low quality 3D reconstruction, on the other hand, could lead to erroneous grasping and hence damaging the produce, leading to unwanted quality degradation.

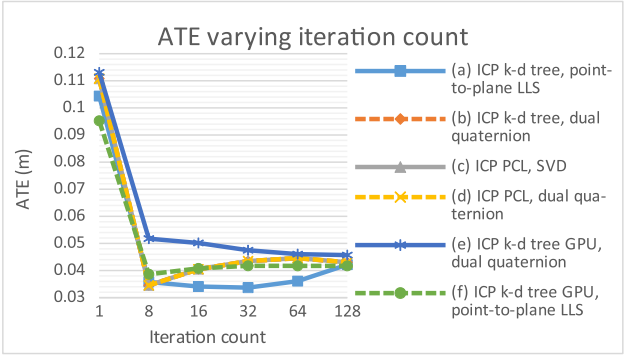
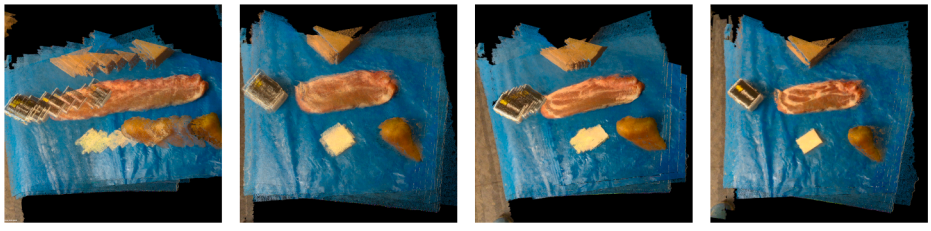


Fig. 9. The absolute trajectory error (ATE) of the data of Fig. 6, varying the iteration count. The point-to-plane LLS transformation estimators converge quickly after a few iterations while the dual-quaternion transformation estimators require more iterations to converge to the same error value. Because of measurement and movement errors as described in Section 4.7, the optimal registration for the fruits gives an error at about 4 cm, as shown in Table 1, and is why the error is increasing with more iterations. The main observation is that the ATE converge to a common point, which results in registrations as shown in Figs. 1 and 2. Line (c) is hidden behind (d).



(a) SVD - Without correspondence rejection

(b) SVD - With correspondence rejection

(c) LLS - Without correspondence rejection

(d) LLS - With correspondence rejection

Fig. 8. Comparing figures a and b to c and d, the PtP LLS transformation estimation is more precise than the standard SVD transformation estimation given a limited number of iterations. Figures b and d shows that the correspondence rejection box from Section 3.4 results in more precise registrations than their equivalents without point correspondence rejection in a and c. Figure b shows a slightly less precise registration than figure d caused by a greater inaccuracy from each registration, making the final result less precise.

4.7. Precision

Fig. 7 shows that most variants are able to perform registrations at a relatively low frame-rate without much loss of precision, and that using more than 10 images (1.38 FPS) generally gives the same registration error for all methods. As expected, when using few images, the small angle approximation of the LLS optimizer is invalidated causing the extreme outliers when using less than 10 images (see Low (2004), Section 3, Linear approximation for small angle approximation).

A more qualitative approach to the evaluation can be seen in Fig. 8, comparing the point-to-plane LLS transformation estimation to the standard SVD transformation estimation, both with and without the correspondence rejection strategy from Section 3.4. The method in Fig. 8d shows that the pork chop and pear are accurately captured, and the high-resolution 3D reconstruction can be used in different stages in the handling and processing pipeline of the agriculture value chain, involving robotic manipulation or automatic inspection tasks.

Comparing Table 1 and Fig. 9 with the work of Lehnert et al. (2016), we see that the error is in the same range for a similar setup, where the Kinect Fusion (Newcombe et al., 2011) algorithm was used and the translational error was just below 3 cm. It is also worth noting that some error is expected when the RGB-D sensor is moving because the SR300 sensor is based on time multiplexed structured light, as can be seen in (Gumhold and König, 2007).

Choi et al. (2013) compare the relative frame-to-frame error and claim that it is in the range of 5 mm to 15 mm. Their absolute trajectory error was in the range of 2 cm. Additionally, Yousif (Yousif et al., 2014), report a similar evaluation where the error is 10's of centimeters for room scale registration using sparse point clouds. Comparing these reported results to our results in Table 1, it appears they are within the same range, with the difference that the focus of our work has been on generating a 3D reconstruction based on high quality and resolution of the point clouds. This shows that, given a good registration algorithm, it is only the precision of the RGB-D sensor that can be a limiting factor.

5. Conclusions

This paper has presented a novel method for the GPU-accelerated, 3D registration of compliant food and other objects using an RGB-D sensor mounted on a 7DoF robot arm, combined with an evaluation of a variety of ICP-based 3D registration algorithms. A GPU implementation of the point-to-plane LLS transformation estimator for the ICP algorithm was applied and evaluated against similar ICP algorithm implementations. The evaluation involved the use of 3D registration tracking, by which the algorithm's generated trajectory was compared to the reference trajectory generated by a robot arm on which an RGB-D camera was mounted. It revealed that the point-to-plane LLS transformation estimator gives the best 3D registration results, both in terms of performance and quality, when validated for compliant food items and produce of agricultural and marine origin. Our implementation consistently outperforms the baselines. Our method shows that adequate registration does not require a high frame rate and this enables real-time registration to be performed at lower RGB-D camera frame rates. We have also demonstrated that GPU implementations of the point-to-plane ICP algorithm can boost 3D registration speeds by up to 33 times compared with CPU implementations for dense point clouds. This provides opportunities for a new range of vision-based, real-world, robotic manipulation applications in the agricultural and marine sectors, where real-time speed and high registration accuracy and resolution are required to enable the correct manipulation of food products without quality degradation.

6. Future work

In our future work we intend to extend the registration algorithm for scan trajectories that do not necessarily have the object centred. This may be achieved by defining the robot pose as an initial pose. We intend to explore a variety of motion scan trajectories, including the use of tailored fixed trajectories for certain kinds of objects, or the analysis of point clouds as the robot moves, involving the use of continuous next-best-view estimates. The work in this paper has focused exclusively on investigations of spatial information in the point cloud. Future work may involve the incorporation of grey-scale or colour values into the registration for correspondence estimation or rejection.

Table 1
Absolute trajectory error (ATE) and Relative frame to frame pose error (RPE) for some of the registrations shown in Figs. 1 and 4. Some error is expected given the precision of the 3D sensor (Carfagni et al., 2017), and the absolute precision of the robot arm. See Sturm et al. (2012) for details about ATE and RPE.

	(a) Apple	(b) Banana	(c) Fruits
ATE	4.27 [cm]	3.29 [cm]	3.80 [cm]
RPE	2.36 [cm]	2.75 [cm]	2.32 [cm]

Supplementary material

For supplementary video, see: https://youtu.be/8E4_hWyL9ZU.

CRediT authorship contribution statement

Ulrich Johan Isachsen: Methodology, Software, Validation, Formal analysis, Investigation, Data curation, Writing - original draft, Writing - review & editing, Visualization. **Theoharis Theoharis:** Writing - original draft, Writing - review & editing, Supervision. **Ekrem Misimi:** Conceptualization, Resources, Writing - original draft, Writing - review & editing, Supervision, Project administration, Funding acquisition.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

We thank NTNU and SINTEF Ocean for providing resources, equipment, and guidance. The financial support from Research Council of Norway through iProcess (255596) and GentleMAN (299757) projects is greatly acknowledged, as is the support for the Open Access publication from NTNU.

References

Agravante, D.J., Chaumette, F., 2017. Active vision for pose estimation applied to singularity avoidance in visual servoing. In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2947–2952. https://doi.org/10.1109/IROS.2017.8206129.

Besl, P.J., McKay, N.D., 1992. A method for registration of 3-d shapes. IEEE Trans. Pattern Anal. Mach. Intell. 14 (2), 239–256. https://doi.org/10.1109/34.121791.

Calli, B., Dollar, A.M., 2016. Vision-based precision manipulation with underactuated hands: Simple and effective solutions for dexterity. In: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1012–1018. https://doi.org/10.1109/IROS.2016.7759173.

Carfagni, M., Furferi, R., Governi, L., Servi, M., Uccheddu, F., Volpe, Y., 2017. On the performance of the intel sr300 depth camera: Metrological and critical characterization. IEEE Sens. J. 17 (14), 4508–4519. https://doi.org/10.1109/JSEN.2017.2703829.

Chen, L.-C., Hoang, D.-C., Lin, H.-I., Nguyen, T.-H., 2016. A 3-d point clouds scanning and registration methodology for automatic object digitization. Smart Sci. 4 (1), 1–7. https://doi.org/10.1080/23080477.2016.1145459.

Chen, Y., Medioni, G., 1991. Object modeling by registration of multiple range images. In: Proceedings. 1991 IEEE International Conference on Robotics and Automation, vol. 3, pp. 2724–2729. https://doi.org/10.1109/ROBOT.1991.132043.

Cherubini, A., Leitner, J., Ortenzi, V., Corke, P., 2018. Towards vision-based manipulation of plastic materials. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 485–490. https://doi.org/10.1109/IROS.2018.8594108.

Choi, C., Trevor, A.J.B., Christensen, H.I., 2013. Rgb-d edge detection and edge-based registration. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1568–1575. https://doi.org/10.1109/IROS.2013.6696558.

Fitzgibbon, A., 2002. Robust registration of 2d and 3d point sets. Image Vis. Comput. 21, 1145–1153. https://doi.org/10.1016/j.imavis.2003.09.004.

Gumhold, S., König, S., 2007. Image-based motion compensation for structured light scanning of dynamic surfaces. Int. J. Intell. Syst. Technol. Appl. (IJISTA) 5, 434–441. https://doi.org/10.1504/IJISTA.2008.021306.

Holz, D., Ichim, A.E., Tombari, F., Rusu, R.B., Behnke, S., 2015. Registration with the point cloud library: A modular framework for aligning in 3-d. IEEE Roboti. Autom. Mag. 22 (4), 110–124. https://doi.org/10.1109/MRA.2015.2432331.

Hu, L., Nooshabadi, S., Ahmadi, M., 2015. Massively parallel kd-tree construction and nearest neighbor search algorithms. In: 2015 IEEE International Symposium on Circuits and Systems (ISCAS), pp. 2752–2755. https://doi.org/10.1109/ISCAS.2015.7169256.

Lehnert, C., Sa, I., McCool, C., Upcroft, B., Perez, T., 2016. Sweet pepper pose detection and grasping for automated crop harvesting. In: 2016 IEEE International Conference on Robotics and Automation (ICRA), pp. 2428–2434. https://doi.org/10.1109/ICRA.2016.7487394.

Low, K.-L., 2004. Linear least-squares optimization for point-to-plane icp surface registration. Tech. Rep. TR04-004. Department of Computer Science, University of North Carolina at Chapel Hill.

Misimi, E., Øye, E.R., Eilertsen, A., Mathiassen, J.R., Åsebø, O.B., Gjerstad, T., Buljo, J., Skotheim, Ø., 2016. Gribbot - robotic 3d vision-guided harvesting of chicken fillets. Comput. Electron. Agric. 121, 84–100. https://doi.org/10.1016/j.compag.2015.11.021.

Misimi, E., Olofsson, A., Eilertsen, A., Øye, E.R., Mathiassen, J.R., 2018. Robotic handling of compliant food objects by robust learning from demonstration. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 6972–6979. https://doi.org/10.1109/IROS.2018.8594368.

Morell-Gimenez, V., Saval-Calvo, M., Azorin-Lopez, J., Garcia-Rodriguez, J., Cazorla, M., Orts-Escolano, S., Fuster-Guillo, A., 2014. A comparative study of registration methods for rgb-d video of static scenes. Sensors 14 (5), 8547–8576. https://doi.org/10.3390/s140508547.

Newcombe, R.A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A.J., Kohi, P., Shotton, J., Hodges, S., Fitzgibbon, A., 2011. Kinectfusion: Real-time dense surface mapping and tracking. In: 2011 10th IEEE International Symposium on Mixed and Augmented Reality, pp. 127–136. https://doi.org/10.1109/ISMAR.2011.6092378.

Pedersen, O.-M., Misimi, E., Chaumette, F., 2020. Grasping unknown objects by coupling deep reinforcement learning, generative adversarial networks, and visual servoing. In: ICRA'20 - IEEE International Conference on Robotics and Automation, Paris, France. https://doi.org/10.1109/ICRA40945.2020.9197196.

Pratikakis, I., Barillot, C., Hellier, P., Mémin, É., 2003. Robust multiscale deformable registration of 3d ultrasound images. Int. J. Image Graph. 3 https://doi.org/10.1142/S0219467803001184.

Qiu, D., May, S., Nüchter, A., 2009. Gpu-accelerated nearest neighbor search for 3d registration. In: Fritz, M., Schiele, B., Piater, J.H. (Eds.), Computer Vision Systems. Springer, Berlin, Heidelberg, pp. 194–203. https://doi.org/10.1007/978-3-642-04667-4_20.

Rusinkiewicz, S., Levoy, M., 2001. Efficient variants of the icp algorithm. In: Proceedings Third International Conference on 3-D Digital Imaging and Modeling, pp. 145–152. https://doi.org/10.1109/IM.2001.924423.

Rusu, R.B., 2009. Semantic 3d object maps for everyday manipulation in human living environments (Ph.D. thesis). Technischen Universität München, Arcisstr. 21 D-80333 München (7 2009).

Sturm, J., Engelhard, N., Endres, F., Burgard, W., Cremers, D., 2012. A benchmark for the evaluation of rgb-d slam systems. In: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 573–580. https://doi.org/10.1109/IROS.2012.6385773.

Yousif, K., Bab-Hadiashar, A., Hoseinnezhad, R., 2014. Real-time rgb-d registration and mapping in texture-less environments using ranked order statistics. In: 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2654–2660. https://doi.org/10.1109/IROS.2014.6942925.

Yuan, W., Mo, Y., Wang, S., Adelson, E.H., 2018. Active clothing material perception using tactile sensing and deep learning. In: 2018 IEEE International Conference on Robotics and Automation (ICRA), pp. 1–8. https://doi.org/10.1109/ICRA.2018.8461164.

Zhou, Q.-Y., Park, J., Koltun, V., 2016. Fast global registration. In: Computer Vision – ECCV 2016. Springer International Publishing, pp. 766–782. https://doi.org/10.1007/978-3-319-46475-6_47.