Estimating the mass of an object from its point cloud for Tactile Internet

Thomas Baars
Supervisor(s): Kees Kroep, Dr. RangaRao Venkatesha Prasad
EEMCS, Delft University of Technology, The Netherlands

23-6-2022

A Dissertation Submitted to EEMCS faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering

Abstract

A problem faced by Tactile Internet (TI) is the distance limitation. A possible solution is to create a simulation of the remote environment, such that the delay experienced by a human operator is reduced. However, to obtain a model of the remote environment, before the user interacts with it, we need to be able to estimate the physical properties of objects in the environment from visual information. This research paper investigates estimating an object's mass from visual information in the form of a point cloud. The object's volume is estimated by surfacing the mesh and dividing the resulting mesh into tetrahedrons, the volume of each tetrahedron can quickly be computed and summed to give an estimate of the total volume. The overall performance is better than the naive approach of using an OBB, however, it is less accurate than a volume slicing approach.

1 Introduction

Tactile Internet (TI) is an emerging technology that will revolutionize various industries [1]. The IEEE P1918.1 (Tactile Internet) standards working group defines TI as "a network (or network of networks) for remotely accessing, perceiving, manipulating, or controlling real or virtual objects or processes in perceived real-time by humans or machines" [2, p. 258]; and give several use cases including teleoperation, automotive, and immersive virtual reality.

Teleoperation allows a human operator to remotely control or manipulate an environment in real-time as if they were physically present [3]. Although not limited to teleoperation, an important aspect of TI is its potential to facilitate teleoperation applications, such as telesurgery, remote disaster management, and telerepair; which allows for the transportation of skills, wherein a skilled worker, like a surgeon or engineer, can operate in remote environments that would otherwise be inaccessible due to safety concerns or logistics [1].

A typical TI system compromises a master domain, controlled domain, and network domain, as depicted in Fig. 1. For example, in telesurgery, where a surgeon (operator) performs a medical procedure by controlling a remote robotic device (teleoperator); then the operator and teleoperator, combined with their interfaces, are referred to as the master domain and controlled domain, respectively [3]. The network domain, or communication network, is used for exchanging the haptic signals (position, texture, force, etc.), video signals, and audio signals between the master and controlled domains [4].

Such a communication network has stricter requirements than one which only exchanges audiovisual signals. In particular, stability and ultra-low latency are needed to achieve fine sensorimotor control. For example, consider the remote balancing of a basketball, as shown in Fig. 2; this requires fast and reliable information about the ball's position, for the operator to understand the situation, react, and for that reaction to arrive at the other end before the ball passes a point of no return and falls [2].

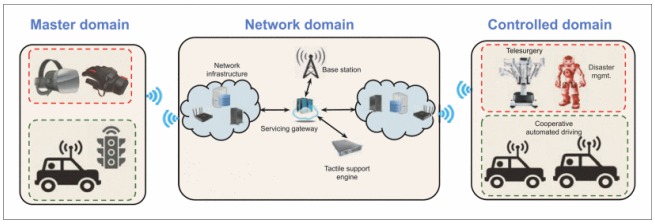


Figure 1: A high-level representation of TI [3].

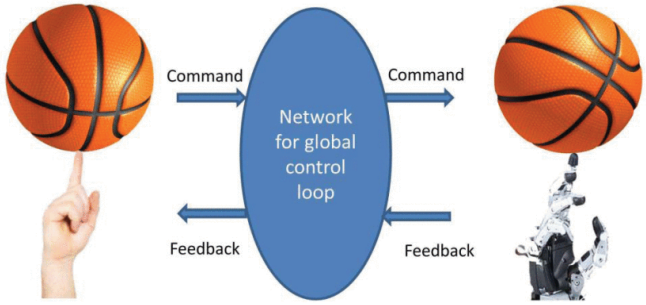


Figure 2: A stable and ultra-low latency network is needed for a human to remotely balance a basketball on a robotic hand. [2]

For such highly dynamic environments, where the exchange of haptic signals is extremely time-critical, a sub-$10\,\mathrm{ms}$ end-to-end (E2E) round-trip latency is needed; but simple propagation delay implies that the endpoints can be a maximum of only $150\,\mathrm{km}$ apart (or only $100\,\mathrm{km}$ apart for propagation through fiber) [2]. A reduction in delay, by increasing the speed by which information can travel through the communication medium, is limited by the speed of light. However, there are other approaches to reducing the delay; for example, Mondal et al. [5] attempts to loosen the E2E distance limitation by forecasting and pre-empting haptic feedback transmission.

A possible alternative solution involves simulation at each endpoint, as outlined in Fig. 3; where communication largely occurs between the master device (M) and a local simulation of the remote environment (C'), and similarly, between the remote controlled device (C) and a local simulation of the master (M'). Meaning the E2E communication network only needs to be used for an occasional synchronization, between an endpoint and its simulation, to ensure the simulated view matches reality. Thus, not only is the delay experienced by a human operator reduced as all perceived interactions occur within a local simulation but the amount of information sent over the network is reduced to just what is needed for synchronization.

However, for this to work, the remote environment needs to be accurately simulated, which involves acquiring knowledge of the physical properties (i.e., mass, center of mass, material) of objects in the scene before the user interacts with them. The proposed solution is to estimate the physical properties of objects using visual information, much like how humans process new environments.

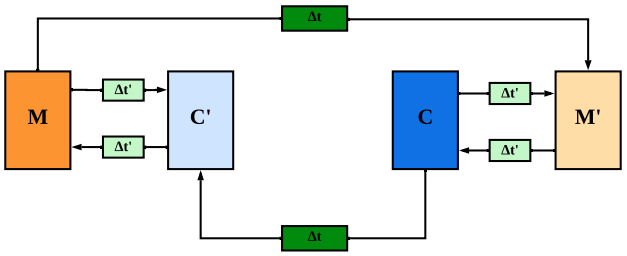


Figure 3: A high-level overview of a possible solution, to the distance limitation of TI, where an operator (M) interacts with a local simulation (C') of the remote environment (C).

Affordable depth sensors, such as the Kinect, give access to high-quality 3D representations of the world, using color and depth information in the form of point clouds [6]. A point cloud is a set of 3D-coordinates, and optionally RGB-values, of the surface of the scanned objects; from which it may be possible to estimate the physical properties of the objects.

The mass of an object is an important physical property [7], and is necessary for the accurate behavior of objects in the simulation. Thus this paper investigates estimating an object's mass from its point cloud. With a focus on estimating the object's volume based on a mesh representation of the point cloud.

The rest of this paper is organized as follows: Section 2 describes existing works. Section 3 gives a formal description of the problems and approaches used to solve them. An implementation of the proposed solution is described in Section 4. The results presented in Section 5, and discussed in Section 6 whereas the ethics and reproducibility of the study are discussed in Section 7. Finally, conclusions are given in Section 8 , along with possible future research in Section 9.

2 Related Works

Recent advances have opened the door to explore the possibility of estimating an object's mass using visual information [8]. For example, there are approaches for specific classes of objects, such as fish [9], beef [10], and pigs [11]; as well as class agnostic approaches, such as [7], which do not rely on class information.

Furthermore, an object's mass is equal to the product of its volume and density, so the problem can be split into the sub-problems of estimating volume and density from visual information. There are many methods for calculating the volume of an object from its point cloud, including those based on volume-intersection methods [12] and variations on slicing methods [13; 14; 15; 16]. Alternatively, the point cloud could be converted into a mesh, using any of the commonly used surfacing methods [17; 18; 19; 20; 21], from which it is possible to calculate its volume.

A promising approach to estimating the density of an object is through the use of active thermography [8].

3 Mass estimation in TI

The goal of this research was to assist with the implementation of the proposed solution to the distance limitation faced by TI, as depicted in Fig. 3. With a focus on moving toward the creation of a simple demonstration, such as a collection of common household objects on a table that a human operator may grab via a remote robotic arm, to test the feasibility of the solution. For this demonstration, it is assumed that there is an RGB-D camera with a clear view of the objects on the table. From this sensory data, we want to create a model of the environment that is as complete as possible. The human operator interacts with a local simulation based on this model, meaning the delay experienced is somewhat decoupled from the network delay, allowing them to perform their tasks smoothly.

To get an accurate model of the environment, we need to be able to estimate the objects' physical properties from the sensory data. There are many challenges to be considered. One challenge is how to separate the objects in the scene, as the sensory data does not distinguish individual objects. Another challenge is handling missing information, as an object may be occluded for numerous reasons: the back face of an object may be missing from a partial view, another object may be blocking the view, the object may have self-occluding geometry, or the object may be in shadow.

In this paper, we investigated the estimation of an object's mass from sensory data. For this, we assumed that some of the aforementioned challenges can and have been solved. Namely, we worked from point clouds of individual objects, which assumes that an object's point cloud can be segmented from the sensory data of the scene. This is a fair assumption as many existing segmentation methods could be used [22]. Another assumption made is that the point cloud does not contain noise, which is a strong assumption depending on the RGB-D camera used, for example in the case of a consumer-grade camera like the Microsoft Kinect V1, but the noise may be reduced by outlier removal based on statistical filtering [6].

We also assumed that the point cloud is not missing information; which is a strong assumption to make, but could be achievable in an environment with multiple RGB-D cameras, ideal lighting, and properly spaced objects, such that the number of partial or obstructed views are reduced. Alternatively, it may be possible to estimate the missing information, for example by mirroring known points along an object's axis of symmetry [23].

Finally, we also assumed that it would be possible to estimate the material characteristics, including the density, of an object from the sensory data. Again, this is a strong assumption, especially when limiting the sensory data to RGB-D values [24]. However, this assumption becomes more realistic when the RGB-D values are supported by supplemental sensory data, such as IR measurements.

Given these assumptions, the initial approach considered was to use object classification based on the features and material properties extracted from the object's point cloud. Unfortunately, with the limited sensory data, the problem of estimating an object's mass does not reduce to a classification task [7]. Causing us to consider a more generalized machine learning approach, similar to the one used by Standley et al. [7], but this would require a large set of training data that would be very time-consuming to create. Additionally, such machine learning approaches may lead to black box models

as solutions, which we want to avoid because they are difficult to validate and do not offer insights into the problems. Thus, we have opted for a more logical approach to estimating the mass of an object.

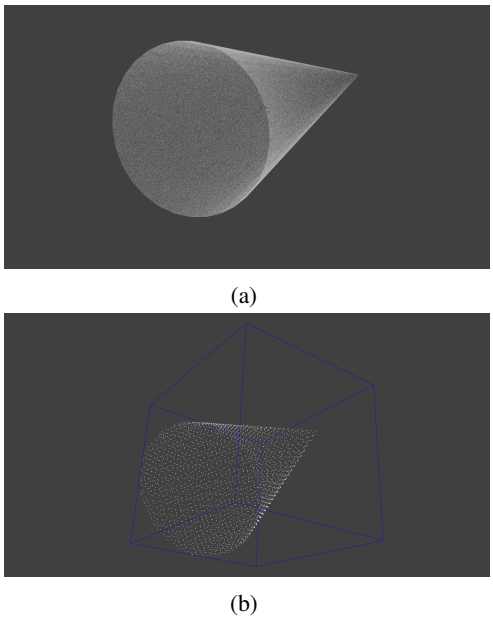
An object's mass is equal to the product of its volume and density, so the task of mass estimation can be divided into estimating an object's volume and density. Since, we have assumed that an object's density can be found as part of the material properties estimated from its point cloud, we focused on estimating an object's volume from its point cloud. Of which there are many existing methods.

A naive approach to volume estimation from a point cloud is based on the volume of its oriented bounding box (OBB). Such an approach is useful for quickly obtaining an upper-bound of the volume estimate, but is generally an over-estimate because it is not a tight fit of the object's surface, as shown in Fig. 4b.

Another approach is to divide the point cloud into slices, compute the volume of each slide, and then sum the volumes to obtain an estimate for the total volume. This slicing approach has many variations, which differ in how the point cloud is sliced and how the volumes of each slice are computed. For example, Zhi et al. [13] cuts the point cloud along Z-axis, then each slice is projected onto the XY plane, from which the data points are filtered using a scanning algorithm to get the area of each slice, and thus the volume of the point cloud. Similarly, Chang et al. [15] cuts the point cloud into slices of equal thickness along the z-axis, bisects each slice along the y-axis, and divides each slice into sub-intervals along the x-axis, then a curve-fitting method is used to estimate the slice contour from which its area can be estimated. Contrasted to Ruchay et al. [16], who make use of a voxel representation of the point cloud and slice method, where an octree is used to generate a three-dimensional density matrix that is sliced into layers along the Z-axis, then the boundary of each layer can be found, using a modified fill algorithm, from which the number of voxels inside the point cloud can be counted and multiplied by the volume of a voxel to give the total volume.

Finally, an alternative approach is to create a mesh, by using a surfacing method, and similarly to the slicing approach, divide the mesh into smaller and easier to compute volumes. Although transforming a point cloud into a mesh may be time-consuming [16], we have still chosen this as the approach to use. This choice was primarily motivated by the fact that the mesh could be used at other points in the pipeline, such as for collision detection.

There are many existing surfacing methods, whose performances depend upon the assumptions that could be made about the input point cloud data such as its level of noise, available information (RGB, normals, intensity), and the number of gaps [18]. Based on our assumptions on our expected point cloud data, we choose to use a surface reconstruction method, as described by Marton et al. [25], which makes use of a greedy triangulation algorithm; that "selects a starting triangle's vertices and connects new triangles until either all points are considered, or no more valid triangles can be connected" [25, 1]. This surfacing method is robust to noise and maintains a list of the boundary points so the mesh can easily be extended when new points are added, for example when a new view of an object is registered with the existing views. This allows us to relax some of the assumptions made, when conducting future research.



(a)

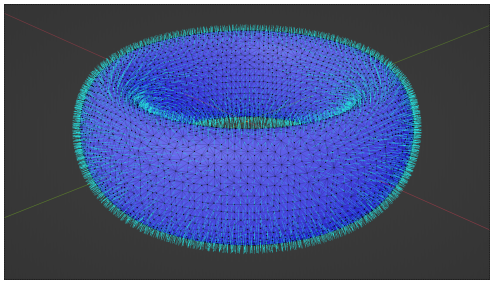
(b)

Figure 4: The top figure shows the point cloud generate by sampling a cone object. The bottom figure shows the downsampled point cloud, contained in the computed OBB depicted in light blue.

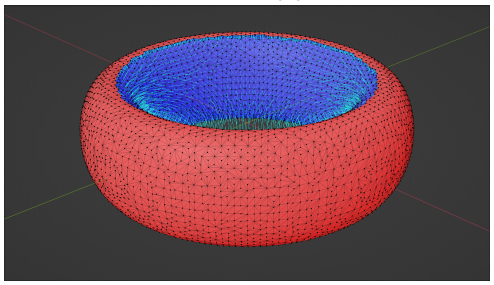
Before applying the surfacing method, we downsample the point cloud to reduce the number of points to a more workable amount, while still trying to maintain the shape of the object; for example, Fig. 4a shows the original point cloud of a cone object and Fig. 4b shows the downsampled point cloud.

After downsampling, we initially estimated the normals of the point cloud based on the coordinates of the points. However, for an accurate estimation of the normals, we required the position of the viewpoint, but this was missing from our test data, because the point clouds were synthesized from multiple viewpoints simultaneously. This meant that when applying the surfacing method, on the point cloud with erroneous normals, we would obtain a mesh where some of the faces have incorrect normals. For example, Fig 5b is the result of the surfacing method applied to the point cloud of a torus object with estimated normals; where faces erroneously facing inside the object are indicated with red, and faces correctly facing outside the object are indicated with blue. The erroneous faces are a result of inaccurate normals, that have erroneously been oriented towards the viewpoint, which by default is the origin. Thus, we instead make use of the computed normals from the same program used to synthesize the point clouds. For example, Fig 5a shows the output of the surfacing method applied to the point cloud with actual normals, which gives the expected result of all faces correctly facing outside the object.

Using the resulting mesh, which consists of a set of triangular faces that make up the surface of the object, we use a similar method to the one used by Zhang et al. [26] to com-



(a) Surfaced torus using given normals.



(b) Surfaced torus using estimated normals.

Figure 5: Highlights the difference between using the greedy triangulation surfacing algorithm on a point cloud with normals and a point cloud with estimated normals. The light blue lines point in the direction of the face's normal. The orientation of the faces are further distinguished by color, where interior faces are colored red and exterior faces are colored blue.

pute the volume of the mesh. Where the mesh is divided into tetrahedrons, the signed volume for each tetrahedron is computed, and all the signed volumes are summed to give the total volume of the mesh.

Instead of being relative to the origin, we compute the signed volumes of the tetrahedrons relative to the center of the computed OBB, such that the triangular face forms the base of the tetrahedron and the center of the OBB is the vertex at the pinnacle of each tetrahedron. For example, in Fig. 6, let v_1, v_2, and v_3 be vertices that form a triangular face in the reconstructed surface and let v_0 be the center of the computed OBB; then the volume of the tetrahedron can be computed as follows:

$$\frac{1}{6}|-(x_2-x_0)(y_3-y_0)(z_1-z_0)+(x_3-x_0)(y_2-y_0)(z_1-z_0) \\ +(x_2-x_0)(y_1-y_0)(z_3-z_0)-(x_1-x_0)(y_2-y_0)(z_3-z_0) \\ -(x_3-x_0)(y_1-y_0)(z_2-z_0)+(x_1-x_0)(y_3-y_0)(z_2-z_0)|$$

Then, to get the sign of the volume, we calculate the inner product of the vector from the center, v_0, to the vertex v_1 and the normal of the triangular face; which is why the normals of the faces must be facing outside the object, where appropriate.

We evaluated the volume estimation approach based on its performance on synthetic point clouds generated from meshes, listed in Fig. 7, which includes simple objects such as a box, cylinder, and cone, that have also been combined to create more complex shapes such as a stack, chunk, and bridge, for a more complete evaluation. To further extend

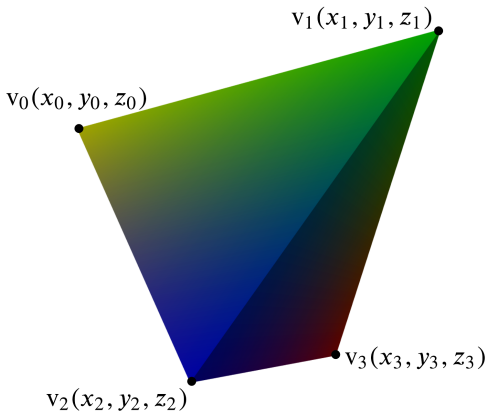


Figure 6: A tetrahedron can be created, where the vertices v_1, v_2, and v_3 form a triangular face in the reconstructed surface and v_0 is the center of the computed OBB.

the test set, we evaluated the approach on rotated versions of these meshes.

4 Implementation Details

The Point Cloud Library (PCL), which is a library for point cloud processing [6], offers many of the methods needed to implement the volume estimation solution. Starting with the VoxelGrid class[1] which can be used for the downsampling of the input point cloud data. The voxelized grid approach downsamples the given point cloud by dividing it into voxels of the specified dimensions and reducing the number of points by approximating it with the centroid of the points contained within the voxel [27].

From the downsampled cloud, the MomentOfInertiaEstimation class[2] can be used to estimate the oriented bounding box (OBB) and center of mass of the object. Based on the vertices of the OBB, it is possible to find its width, height, and depth, from which it is possible to obtain an upper bound on the estimate of the object's volume.

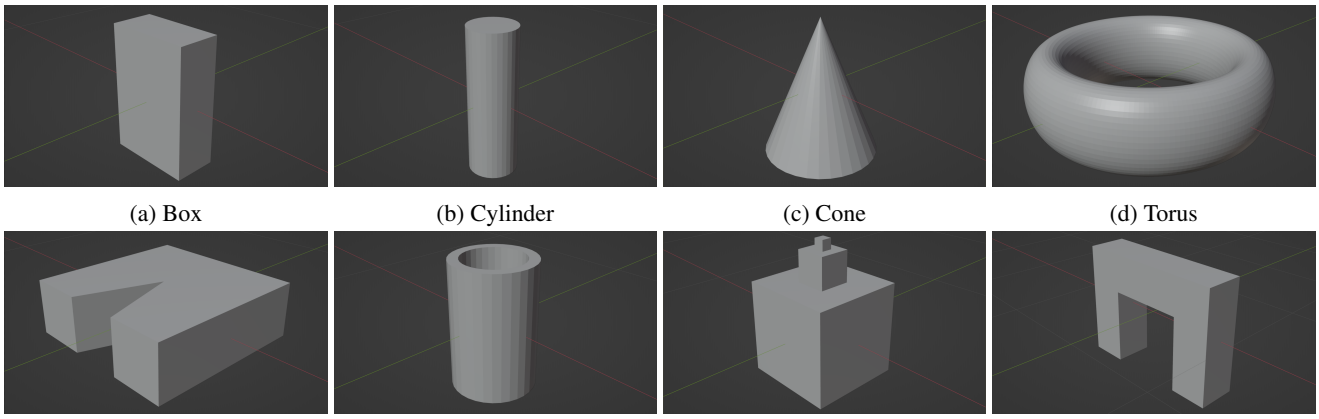
Additionally, the downsampled cloud can be used in the surface reconstruction method. The GreedyProjectionTriangulation class[3] can be used to apply the greedy triangulation surface reconstruction algorithm on the point cloud. A set of triangular faces is returned, in the form of a 3-tuple of indices that can be used to identify points in the point cloud, which make up the surfaced mesh.

Iterating through this set of triangular faces, such that for each face, its vertices and the center of mass form a tetrahedron, for which the signed volume is calculated and then added to the running total volume of the mesh. Once all triangular faces have been processed, the total volume gives an estimate of the object's volume.

[1] http://pointclouds.org/documentation/classpcl_1_1_voxel_grid.html

[2] http://pointclouds.org/documentation/classpcl_1_1_moment_of_inertia_estimation.html

[3] http://pointclouds.org/documentation/classpcl_1_1_greedy_projection_triangulation.html



(a) Box (b) Cylinder (c) Cone (d) Torus

(e) Chunk (f) Mug (g) Stack (h) Bridge

Figure 7: The objects used to generate synthetic point clouds for the evaluation of the estimation approach.

For the evaluation of the estimation method, a test set was created by synthesizing point clouds from artificial objects with known ground truth values. These artificial objects were created from mesh primitives in Blender[4], as depicted in Fig. 7. Copies of these meshes were made, which were then rotated along the Z-axis, then the local X-axis, and finally the local Y-axis, by 7 and 45 degrees. The faces of all these meshes were converted to triangular faces, and exported as OBJ files, so they could be used as inputs to the volume estimation method to determine their expected values.

CloudCompare[5], which is a 3D point cloud and mesh processing software, was used to create 100 different random samples, with each sample containing approximately a million points consisting of XYZ-data and normals, based on the surface of each mesh. The 100 samples are then used to get the mean volume estimate, using the naive OBB approach and the surfacing approach, which can then be compared to the expected value for each mesh.

5 Results

Table. 1 contains the results of running the oriented bounding box (OBB) approach to volume estimation on the 100 random samples of each mesh, depicted in Fig. 7, rotated by 0, 7, and 45 degrees (along the z-axis, followed by the same rotation along the local x-axis, and again along the local y-axis). Where the "Expected Value" represents the ground truth, which has been calculated by dividing the artificial object into tetrahedrons and summing their signed volumes, the "Mean OBB Estimate" represents the average of the computed volumes, using the OBB approach, over the 100 samples for that object at the specified rotation, and the error is given by the average absolute difference between the expected and actual values by "Mean OBB Error" as well as by a percentage of the expected value by "Mean OBB Error (%)". The values in the table are rounded to two decimal places to make it more readable. Similarly, Table. 2 shows the same samples except their volumes are estimated using the surfacing approach. The mass estimate scales proportional to the volume estimate, as it is computed using the density value from the material properties estimate multiplied by the volume estimate.

[4]https://www.blender.org/

[5]http://cloudcompare.org/

6 Discussion

Based on the results in Tables 1 and 2, we can see that the surfacing approach to volume estimation generally performs better, than the naive oriented bounding box (OBB) approach to volume estimation, as it generally has a lower mean percentage error. This is expected as the surface approach generally gives a better fit of the underlying object. However, it does not perform as well as some other existing methods for volume estimation from point clouds; for example, Chang et al. [15] managed to achieve a consistent sub-1% error.

One possible explanation for the approach's poor performance is an incompatible test set. That is, the volume is estimated based on the constructed surface using an algorithm that assumes "the deviation between the normals of any two incident triangles on a vertex is less than 90°" [25, 1]. This is not the case for the test set which makes use of computed normals, that are at sharper angles than if the normals were estimated. Resulting in holes in the constructed surface, for example, see Fig 8, which are not included in the volume estimation. It is possible to change the maximum surface angle considered by the surfacing algorithm, however a value that performs well on one mesh may not perform well on another; for example, the inner geometry of the chunk mesh may be erroneously joined when using a maximum surface angle.

Alternatively, we could synthesize the point clouds for the test set by obtaining the complete view of the object by sequentially capturing partial views, estimating the normals based on the current viewpoint, and then registering the results to give a complete point cloud with smoother normals. Such an approach mirrors the one used to obtain a complete point cloud in real-life. However, there are still other parameters, such as the voxel leaf size, which affect the performance of the surface-based volume estimation approach for certain object shapes. Similarly, volume slicing approaches, like the one used by Chang et al. [15], also rely on finding the optimal

Object	Rotation (Degrees)	Expected Value (cm^3)	Mean OBB Estimate (cm^3)	Mean OBB Error (cm^3)	Mean OBB Error (%)
Box	0	20,250.00	20,253.93	3.93	0.02
	7	20,250.00	20,857.21	607.21	3.00
	45	20,250.00	20,506.21	256.21	1.27
Cylinder	0	7,901.16	10,055.39	2,154.23	27.26
	7	7,901.14	10,188.60	2,287.46	28.95
	45	7,901.18	10,140.96	2,239.78	28.35
Cone	0	4,581.66	17,931.24	13,349.58	291.37
	7	4,581.66	18,093.50	13,511.84	294.91
	45	4,581.66	18,219.28	13,637.62	297.66
Torus	0	10,846.60	28,223.36	17,376.76	160.20
	7	10,846.60	28,319.24	17,472.64	161.09
	45	10,846.60	28,389.51	17,542.91	161.74
Chunk	0	66,000.00	72,585.58	6,585.58	9.98
	7	66,000.00	84,598.15	18,598.15	28.18
	45	65,999.90	84,133.99	18,134.09	27.48
Mug	0	46,941.40	95,572.11	48,630.71	103.60
	7	46,941.50	95,669.36	48,727.86	103.81
	45	46,941.60	96,368.45	49,426.85	105.29
Stack	0	129,803.00	368,331.36	238,528.36	183.76
	7	129,803.00	331,336.29	201,533.29	155.26
	45	129,803.00	188,853.91	59,050.91	45.49
Bridge	0	512,000.00	770,113.65	258,113.65	50.41
	7	512,000.00	770,717.17	258,717.17	50.53
	45	511,999.00	770,656.72	258,657.72	50.52

Table 1: Results based on the oriented bounding box (OBB) approach to volume estimation.

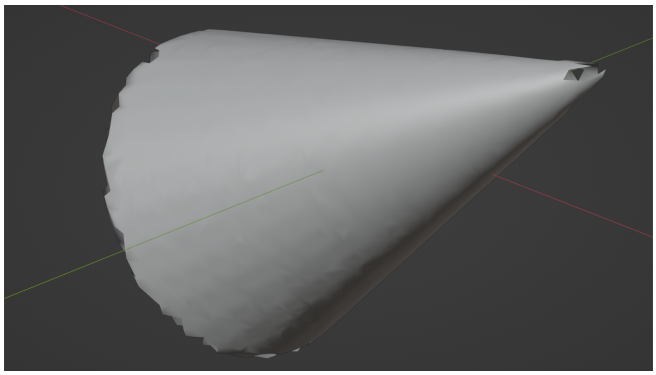


Figure 8: The result of the surfacing algorithm applied to the downsampled point cloud of a cone object, that has been rotated 45 degrees along various axes. The resulting surface contains erroneous holes that may be the cause of the large error in the volume estimate.

parameters for certain objects. Thus, for a more generalizable solution, it may be beneficial to consider a machine learning approach.

Another explanation for the approach's poor performance is the sensitivity, of the implementation of the calculation of the volume of each tetrahedron, to variations in the position of the center point. This is most notable from the discrepancies in the "Expected Values", in Tables 1 and 2, where the expected value varies for the same object with a different rotation, for example, the mug goes from 46941.40 cm^3 to 46941.60 cm^3 when rotated 45 degrees, but the volume should be unaffected by rotations. However another cause of these discrepancies in the "Expected Values" may be a result of floating point errors, as the volumes are being computed in terms of meters cubed, so to represent the artificial objects which are only thousands of centimeters cubed, requires many decimal places; especially when they are being subdivided into small tetrahedrons.

7 Responsible Research

The research was entirely conceptual and did not involve any human participants, so there are not many ethical aspects to consider. However, wherever possible, we made a strong effort to conduct and present our research in a responsible manner. This includes giving credit to the authors of any works that were used, presenting all the data points collected unless their absence could be properly motivated, and finally being objective in our findings.

The reproducibility of the research relates to conducting the research in a responsible manner. A large focus of this study was on the reproducibility of the results, which was one of the motivations for the choice of using a synthetic data set for evaluation. Additionally, open source tools such as Blender, CloudCompare, and the Point Cloud Library (PCL), were chosen to make evaluation easily accessible to anyone. Finally, a thorough description of the set up of the environment used to evaluate the performance of the solutions was

Object	Rotation (Degrees)	Expected Value (cm^3)	Mean Mesh Estimate (cm^3)	Mean Mesh Error (cm^3)	Mean Mesh Error (%)
Box	0	20,250.00	18,629.75	1,620.25	8.00
	7	20,250.00	18,689.14	1,560.86	7.71
	45	20,250.00	19,161.78	1,088.22	5.37
Cylinder	0	7,901.16	7,387.25	513.91	6.50
	7	7,901.14	7,478.40	422.74	5.35
	45	7,901.18	7,496.47	404.71	5.12
Cone	0	4,581.66	4,655.32	73.66	1.61
	7	4,581.66	4,620.15	38.49	0.84
	45	4,581.66	2,605.36	1,976.30	43.13
Torus	0	10,846.60	10,768.54	78.06	0.72
	7	10,846.60	10,763.14	83.46	0.77
	45	10,846.60	10,763.63	82.97	0.76
Chunk	0	66,000.00	61,425.68	4,574.32	6.93
	7	66,000.00	62,475.71	3,524.30	5.34
	45	65,999.90	63,269.08	2,730.82	4.14
Mug	0	46,941.40	45,555.86	1,385.55	2.95
	7	46,941.50	45,377.22	1,564.28	3.33
	45	46,941.60	45,565.43	1,376.17	2.93
Stack	0	129,803.00	123,148.25	6,654.75	5.13
	7	129,803.00	124,166.36	5,636.64	4.34
	45	129,803.00	125,659.88	4,143.12	3.19
Bridge	0	512,000.00	495,681.30	16,318.70	3.19
	7	512,000.00	497,514.15	14,485.85	2.83
	45	511,999.00	500,105.57	11,893.43	2.32

Table 2: Results based on the surfacing approach to volume estimation.

given.

8 Conclusions

Tactile Internet has many applications across multiple fields. One use case for TI is teleoperation, which would allow for efficient transportation of skills. However, in highly dynamic teleoperation applications, an ultra-low latency network is needed for communication, this limits how far endpoints can be from each other. One possible solution to this distance limitation involves simulating the remote environment such that the delay experienced by the human operator is reduced. But to simulate the remote environment, before the user interacts with it, would require existing knowledge of the environment or the capability to obtain a model from visual information.

Thus as part of the research into the feasibility of estimating the physical proprieties of objects, in a remote environment, based on visual information; this research paper focused on the estimation of an object's mass from RGB-D data. Certain assumptions were made about the sensory data, including assuming that, firstly, the data does not contain noise and is complete. Secondly, it is assumed that the objects can be segmented from the sensory data of the scene. Finally, it is assumed that an object's density can be determined from its estimated material properties from the sensory data.

Given these assumptions, an volume estimation approach that uses a greedy triangulation surfacing algorithm was investigated, by evaluating its performance on an artificial test set. From which it was determined that the approach performs better than the naive oriented bound box (OBB) approach to volume estimation, but in terms of percentage error, the approach does not perform as well as other volume estimation approaches that make use of a slicing approach. However it is difficult to compare the reported results; as the authors of the papers, who have investigated such slicing approaches to volume estimation, have not made the point cloud used for their evaluations publicly available. Additionally, the method we used to generate the point clouds used for evaluation may have negatively affected the validity of the results. As we make use of synthetically computed normals, as opposed to estimate normals, which are not as compatible with the chosen surfacing algorithm; this may explain why an artificial object such as the cone rotated by 45 degrees may perform poorly, leading to a 43.13 percent error.

9 Future Work

The point clouds used to evaluate the performance of the estimation approach could be improved by incrementally obtaining various partial views of an artificial object, from which the normals can be estimated based on the current position of the viewpoint, and registered with the current profile, until a complete view of the object is obtained. Not only does this approach resemble how point clouds are obtained in reality, but also means the normals are distributed more smoothly which may lead to better surfacing performance. This improved test set can then be used to properly compare the var-

ious existing methods for volume estimation.

Research should be conducted to determine how realistic are the assumptions that were made. For example, can an object's density be accurately determined from its estimated material properties based on the sensory data? If these assumptions are too strong, then perhaps a more generalizable machine learning approach should be considered for obtaining an object's mass from the sensory data. Finally, research needs to be conducted on what level of accuracy, for the estimated mass, is required to create a realistic simulation.

References

[1] J. P. Verburg, H. J. C. Kroep, V. Gokhale, R. V. Prasad, and V. Rao, "Setting the yardstick: A quantitative metric for effectively measuring tactile internet," in *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, p. 1937–1946, IEEE Press, 2020.

[2] O. Holland, E. Steinbach, R. V. Prasad, Q. Liu, Z. Dawy, A. Aijaz, N. Pappas, K. Chandra, V. S. Rao, S. Oteafy, M. Eid, M. Luden, A. Bhardwaj, X. Liu, J. Sachs, and J. Araujo, "The ieee 1918.1 "tactile internet" standards working group and its standards," *Proceedings of the IEEE*, vol. 107, pp. 256–279, 2 2019.

[3] V. Gokhale, K. Kroep, V. S. Rao, J. Verburg, and R. Yechangunja, "Tixt: An extensible testbed for tactile internet communication," *IEEE Internet of Things Magazine*, vol. 3, pp. 32–37, 4 2020.

[4] S. K. Sharma, I. Woungang, A. Anpalagan, and S. Chatzinotas, "Toward tactile internet in beyond 5g era: Recent advances, current issues, and future directions," *IEEE Access*, vol. 8, pp. 56948–56991, 2020.

[5] S. Mondal, L. Ruan, and E. Wong, "Remote human-to-machine distance emulation through ai-enhanced servers for tactile internet applications," in *Optical Fiber Communication Conference*, p. M1A.6, OSA, 2020.

[6] R. B. Rusu and S. Cousins, "3d is here: Point cloud library (pcl)," in *2011 IEEE International Conference on Robotics and Automation*, pp. 1–4, IEEE, 5 2011.

[7] T. Standley, O. Sener, D. Chen, and S. Savarese, "image2mass: Estimating the mass of an object from its image," in *Conference on Robot Learning* (S. Levine, V. Vanhoucke, and K. Goldberg, eds.), vol. 78, pp. 324–333, PMLR, 5 2017.

[8] T. Aujeszky, G. Korres, M. Eid, and F. Khorrami, "Estimating weight of unknown objects using active thermography," *Robotics*, vol. 8, 12 2019.

[9] M. O. Balaban, G. F. Ünal Şengör, M. G. Soriano, and E. G. Ruiz, "Using image analysis to predict the weight of alaskan salmon of different species," *Journal of food science*, vol. 75, no. 3, pp. E157–E162, 2010.

[10] Ş. Taşdemir, M. Yakar, A. Ürkmez, and Ş. İnal, "Determination of body measurements of a cow by image analysis," in *Proceedings of the 9th International Conference on Computer Systems and Technologies and Workshop for PhD Students in Computing*, pp. V–8, 2008.

[11] Y. Yang and G. Teng, "Estimating pig weight from 2d images," in *International Conference on Computer and Computing Technologies in Agriculture*, pp. 1471–1474, Springer, 2007.

[12] B. Dellen and I. A. Rojas Jofre, "Volume measurement with a consumer depth camera based on structured infrared light," in *Proceedings of the 16th Catalan Conference on Artificial Intelligence, poster session*, pp. 1–10, 2013.

[13] Y. Zhi, Y. Zhang, H. Chen, K. Yang, and H. Xia, "A method of 3d point cloud volume calculation based on slice method," in *2016 International Conference on Intelligent Control and Computer Application (ICCA 2016)*, pp. 155–158, Atlantis Press, 2016.

[14] B. Li, J. Wei, L. Wang, B. Ma, and M. Xu, "A comparative analysis of two point cloud volume calculation methods," *International Journal of Remote Sensing*, vol. 40, pp. 3227–3246, 4 2019.

[15] W.-C. Chang, C.-H. Wu, Y.-H. Tsai, and W.-Y. Chiu, "Object volume estimation based on 3d point cloud," in *2017 International Automatic Control Conference (CACS)*, pp. 1–5, IEEE, 2017.

[16] A. Ruchay and M. Fedorova, "Fast algorithm of 3d object volume calculation from point cloud," in *Applications of Digital Image Processing XLIV*, vol. 11842, p. 118421Q, International Society for Optics and Photonics, 2021.

[17] M. Kazhdan and H. Hoppe, "Screened poisson surface reconstruction," *ACM Transactions on Graphics (ToG)*, vol. 32, no. 3, pp. 1–13, 2013.

[18] M. Berger, A. Tagliasacchi, L. Seversky, P. Alliez, J. Levine, A. Sharf, and C. Silva, "State of the art in surface reconstruction from point clouds," in *Eurographics 2014-State of the Art Reports*, vol. 1, pp. 161–185, 2014.

[19] K. Zhou, M. Gong, X. Huang, and B. Guo, "Data-parallel octrees for surface reconstruction," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, pp. 669–681, 2011.

[20] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3d surface construction algorithm," *ACM siggraph computer graphics*, vol. 21, no. 4, pp. 163–169, 1987.

[21] G. Roth and E. Wibowoo, "An efficient volumetric method for building closed triangular meshes from 3-d image and point data," in *Graphics Interface*, 1997.

[22] E. Grilli, F. Menna, and F. Remondino, "A review of point clouds segmentation and classification algorithms," *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 42, p. 339, 2017.

[23] T. Buijs, "3d point cloud completion from 2.5d data," 2022. Pre-print.

[24] H. Yang, "Acquiring material properties of objects for tactile simulation through point cloud scans," 2022. Preprint.

[25] Z. C. Marton, R. B. Rusu, and M. Beetz, "On Fast Surface Reconstruction Methods for Large and Noisy Datasets," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, (Kobe, Japan), May 12-17 2009.

[26] C. Zhang and T. Chen, "Efficient feature extraction for 2d/3d objects in mesh representation," in *Proceedings 2001 International Conference on Image Processing (Cat. No.01CH37205)*, vol. 3, pp. 935–938 vol.3, 2001.

[27] M. Miknis, R. Davies, P. Plassmann, and A. Ware, "Near real-time point cloud processing using the pcl," in *2015 International Conference on Systems, Signals and Image Processing (IWSSIP)*, pp. 153–156, 2015.