Simple and Effective Load Volume Estimation in Moving Trucks using LiDARs

Lucas L. Amorim*, Filipe Mutz*†, Alberto F. De Souza*, Claudine Badue* and Thiago Oliveira-Santos*
*Universidade Federal do Espírito Santo (UFES), Vitória, ES, Brazil
†Instituto Federal do Espírito Santo (IFES), Serra, ES, Brazil
*Email: lucasluppiam@gmail.com

***Abstract*—Industries need to track the amount of materials and goods transported through processing units in order to optimize production. In large-scale industries, trucks and trains are commonly used for transportation. The manual evaluation of the volume of material being transported by these vehicles can be imprecise, inefficient, and even unsafe for employees. Therefore, this work presents an automated system for estimating the volume of load in moving trucks using a pair of multi-layer light detection and ranging (LiDAR) sensors. The sensors are mounted in a structure so that trucks can pass through without stopping. The proposed system can be used with any type of compact load such as grains, and powders. A mesh of the load is built and used for estimating the volume. A simple, efficient, and effective heuristic is proposed for tracking the truck's positions. The system was deployed and evaluated in a mining company in real conditions of operation. Experimental results indicate that the system produces accurate estimates of the volume of ore powder transported by trucks. The reconstruction of the loads and the estimative of their volumes are performed once the data is acquired and lasts less than 1.5 minutes on average.**

I. Introduction

In order to maximize profit, industries have to optimize their processes and achieve high throughput of products. By measuring the amount of materials and goods transported through processing units, managers can schedule the use of machinery, and plan ahead to minimize delays and stall times.

In large-scale production environments such as agriculture and livestock farms, mining industries, and wood extraction companies, trucks or trains are commonly used for transportation. Measuring the load weight and/or its volume are ways of quantifying the amount of material being carried by these vehicles. Large and potentially expensive scales are required for weighting the loads. If there are not sufficient scales to supply the demand of incoming vehicles, queues can be formed which is inefficient. Moreover, due to the scales sizes, moving them around in the factory is impractical. Depending on the factory's organization, vehicles may need to take large detours to reach the scales.

An alternative to weighting the loads is the measurement of their volumes. In some situations, the volume of the materials is the primary information needed by the managers, for instance, to know the amount of space required for storing them. Measuring the volume directly can be more precise than estimating it through the weight and the density of the material since the density may vary depending on several factors (e.g., temperature and humidity). However, measuring volume can be challenging, in particular if the surface of the material is irregular. Performing the measurements manually can be inefficient, laborious, and even unsafe for the employees. Automatic methods have the potential to be precise and efficient, and they prevent human-errors, accidents, and drops in productivity, for instance, due to tiredness and distraction.

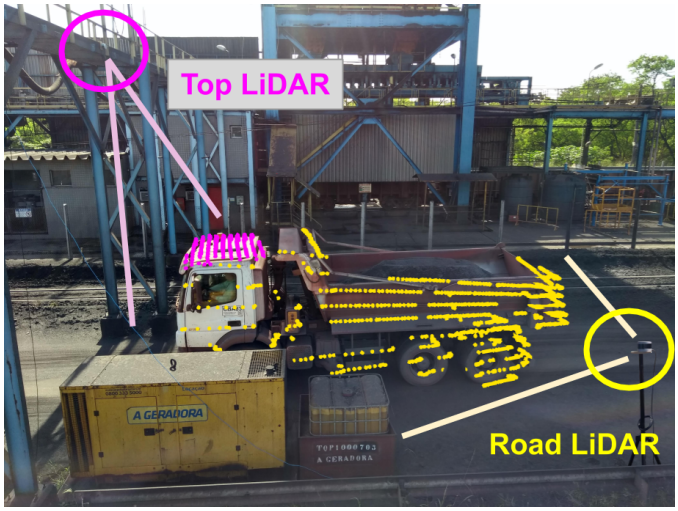

Fig. 1: Illustration of the sensors' setup in the proposed system. The top LiDAR sensor (highlighted in pink) scans the surface of the load from the top and produces the point cloud V_{vol}. It is used for reconstructing the load and estimating its volume. The road LiDAR sensor scans the truck from the side of the road and produces the point cloud V_{pos}. It is used for tracking the truck position as it moves underneath the structure.

Previous works tried and developed automatic methods for measuring the volume of different types of materials. In [1], five cameras and computer vision techniques are used for segmenting logs being transported by a truck and for estimating their volume. A percentual error of 5% in relation to the real volume was achieved. In [2], a multi-layer light detection and ranging sensor (LiDAR) is used for counting the number of logs in the body of a truck. In [3], an unmanned aerial vehicle equipped with a GPS/GLONASS system and a camera was used for measuring the volume of log carried by a truck. The measurements are performed when the truck is not moving using multi-view photogrammetry and 3D reconstruction techniques. They achieved an error of 0.5m^3

or, equivalently, an percentual error of 1.7% in relation to the real volume. In [4], a high-resolution stereo camera is used for estimating the volume of soil in the bucket of an excavator. Stereo matching algorithms are used for computing depth maps. A model of the empty bucket is created before operation. The model is manually split in two parts, the borders and the inner body. The iterative closest point (ICP) algorithm is used for finding the bucket pose during operation and the inner body is used for estimating the volume of soil. They achieved errors ranging from 50cm^3 (15% of the real volume) to 2cm^3 (0.57% of the real volume).

In the works mentioned before, the authors assume the vehicle is not moving during the measurement process. The following works propose methods that allow measuring the volume with the vehicle in motion. In [5], two single-layer LiDARs are used for measuring the volume of dirt being transported by haul-trucks. The lasers are mounted in a structure high above the roadway pointing downwards with orthogonal orientations. The data collected when the haul-trucks pass underneath the structure are used for estimating the volume of the loads. The localization of the haul-trucks is obtained using the ICP algorithm. The technique for estimating the volume of the loads is not presented in the paper (it was under provisional patent). The accuracy achieved by the technique is also not presented. In [6], three LiDARs are used for measuring the volume of logs in moving trucks with percentual errors ranging from 0.1% to 4.2%. The closed software Woodtech Logmeter 4000 is used for measuring the volumes. Since the software is not open, it is hard to predict in which conditions it will operate successfully.

This work presents a system for measuring the volume of compact materials (materials with small rate of empty spaces in relation to the occupied volume) transported by a moving truck using two multi-layer LiDARs positioned as illustrated in Figure 1. LiDAR sensors are becoming very popular and inexpensive due to the demand for autonomous vehicles. The road LiDAR (emitting yellow rays in Figure 1) is mounted aside the road with the central ray parallel to the road. This sensor produces measurements that allow tracking the truck position by means of a simple, effective, and efficient technique that is presented in this work. The top LiDAR (emitting pink rays in Figure 1) is mounted high above the road with the central ray orthogonal to the road. This sensor is used for observing the load surface on the truck bucket. The positions estimated using the road LiDAR are used for integrating the measurements of the top LiDAR in a complete model of the truck. An *a priori* model of the empty truck bucket is used along with the truck point cloud reconstructed from the LiDARs' measurements for creating a mesh of the load. The volume of the load is approximated by the volume of the mesh. The proposed system was deployed and evaluated in a mining company in real conditions of operation. The system was successfully used for measuring the volume of ore powder transported by the trucks. The data is collected very quickly with the truck in movement and the complete process of estimating the volume given the sensors' data takes less than 1.5 minutes on average.

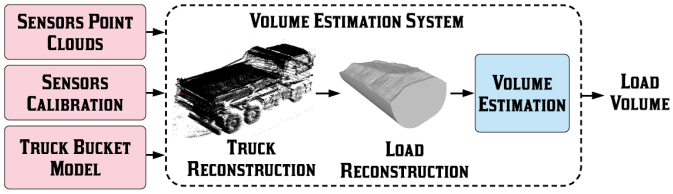

Fig. 2: Overview of the system for estimating the load volume in moving trucks. The system receives as input the data collected by two LiDAR sensors, integrate them into a dense point cloud representing the truck, isolates the points regarding to the load and reconstructs a closed mesh, and, finally, estimates the volume using the closed mesh.

II. Volume Estimation in Moving Trucks

This section describes the hardware setup and the proposed software for measuring the load volume in moving trucks. The system comprises two multi-layer light detection and ranging (LiDAR) sensors, in particular two Velodyne VPL-16, for measuring the volumes. Each Velodynes VPL-16 is composed of 16 lasers assembled along the vertical axis (one above the other). The set of 16 lasers is rotated around the vertical axis by a motor so that the sensor is capable of observing 360 degrees around it. The LiDAR returns a point cloud in which the points are represented in spherical coordinates.

The pair of LiDARs (one placed above and one on the road behind the truck) are mounted in pre-existent structures of the factory, as presented in Figure 1. The top LiDAR is mainly used for reconstructing the load and performing the volume measurement. Therefore, it is mounted in a platform 7m above the road with the central ray orthogonal to the road and with the axis of rotation nearly parallel to the trajectory of the truck. This configuration ensures that each LiDAR rotation will generate many readings (for each laser) from one side to the other of the truck. The road LiDAR is mainly used for estimating the truck position for each LiDAR reading. Therefore, it is mounted aside the road and behind the truck. To ensure the rays hit the back of the truck bucket, the LiDAR is placed 2m above the ground with the central ray parallel to the road and with the rotation axis nearly orthogonal to the ground. The top LiDAR will generate the point cloud V^i_{vol} at every timestamp i and the road LiDAR will generate the point cloud V^j_{pos} at every timestamp j.

Once the sensors are in place, they need to be calibrated so that their relative pose is known. The calibration parameters are later used to map the point clouds across the coordinate systems of the LiDARs. The calibration process only needs to be performed once, since the sensors' positions are fixed. One additional input parameter required by the system is the model of an empty truck bucket. This model comprises two parts, the external part and the internal part. The external part is used for identifying the pose of the bucket in the truck point cloud. The internal part is used for generating the base of the load for volume measurement. The truck bucket model can either be

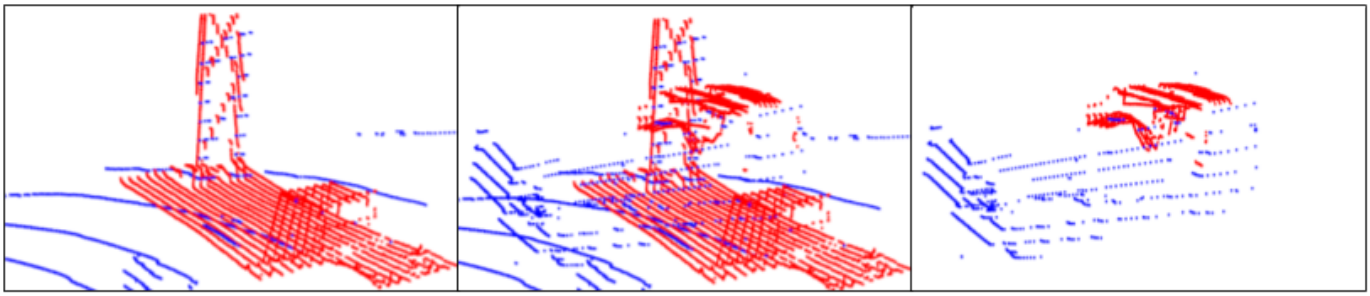

Fig. 3: The left image shows a point clouds of the environment without a truck. The image in the center shows the point cloud with a truck, whereas the image on the right shows the point cloud after removing points outside the region of interest (represented by a box) and points from the background.

provided by the manufacturer or reconstructed by the system using an empty truck (the latter was used in this work).

The system comprises three software modules (see Figure 2): truck reconstruction, load mesh reconstruction, and volume estimation. During the measurements, the truck moves through the region observed by the sensors performing a linear trajectory. The sensors' data are recorded and used as input for the truck reconstruction module. The trucks do not have to stop for the measurement process. The truck reconstruction module is responsible for tracking the current position of the truck and for integrating the LiDARs' point clouds over time in order to produce a dense point cloud of the truck. The load mesh reconstruction module is responsible for identifying the points from the dense point cloud that belongs to the load and for generating a closed mesh representing the region of interest. The external part of the empty bucket model is used in the identification process and the internal part is used for generating the closed mesh. Finally, the volume estimation module approximates the volume of the load as the inner volume of the mesh. These steps are detailed in the following sections.

A. Truck Reconstruction

This module receives the raw data from the LiDAR sensors and reconstructs the dense point cloud representing the truck. The whole process comprises the following steps: temporal data correction, background removal, LiDAR sensors calibration, truck path estimation, truck position estimation, and point cloud coordinate system alignment.

1) Temporal Data Correction: The raw data of each LiDAR is a point cloud comprising the measurements of each ray for one revolution, i.e., from 0 to 360 degrees. The LiDAR used in this work rotates at a frequency of 20Hz and samples about 1808 points for each laser in one revolution. Each point cloud is recorded with an associated timestamp (V^i_{pos} and V^j_{vol} are respectively the point clouds of the timestamp i and j generated by the top and the road LiDARs respectively) as if all points were acquired at that instant. This assumption is not harmful when the environment and the sensors are static. However, because the truck is moving, the raw data have to be corrected to account for the fact that the sensor performs readings as it rotates and the rotations are not instantaneous. This is specially important for the LiDAR used for load reconstruction because the truck will be in one position when laser hits one side of the truck and in another when it hits the other side. In order to reduce such temporal sliding, each point measurement is corrected to the same timestamp (reference timestamp for the point cloud) by using the average speed of the truck and the trajectory path (described in the next subsections). In other words, the points are corrected by a delta shift that is calculated using the current truck speed and the delta time between the first point measured by the sensor and the current point. The output of this step is the point cloud of one LiDAR revolution V^j_{vol} without the sliding error.

2) Background Removal: This step aims at cleaning the scene from points that do not belong to the truck. The very first step is defining a region of interest (a box marking the region where the truck passes) and removing every point that is outside the region. The second step is the actual background removal. Since the scene is mostly static and the LiDARs do not move, a technique similar to image background removal can be applied. During the calibration step, a point cloud of the empty environment is captured using both LiDARs and stored as the background point cloud. During the measurements, the truck is segmented from the environment objects by removing every point from the current point cloud that is sufficiently close to a point from the background point cloud. The euclidean distance is used for measuring the proximity of the points and a KD-Tree structure [7] is used for optimizing the search. Basically, a KD-Tree is built with the points of the background point cloud and the nearest neighbor of each point of the current point cloud that is below a threshold is removed from the current point cloud. The threshold was empirically defined to 10cm, considering a safety distance that would not remove points of interest belonging to the bucket. This process is performed for both LiDAR point clouds V^i_{pos} and V^j_{vol}. The output of this step are cleaner point clouds, V^{i-}_{pos} and V^{j-}_{vol}, for each LiDAR revolution (i.e., without points belonging to the background such as the road). It is worth noting that some nosy measurements caused by dust in the air are still present in the clean point cloud. This process is repeated continuously and it is used as decision for carrying on with the rest of the measurements. When there are enough remaining points after cleaning (at least 1000 points in V^j_{vol}), the system assumes

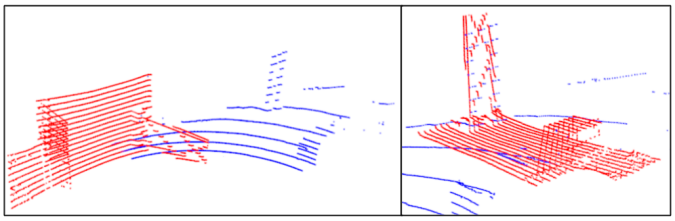

Fig. 4: Illustration of the point cloud projection from the coordinate system of one LiDAR to the other. The left image shows both point cloud without calibration, whereas the right image shows both point cloud after the calibration.

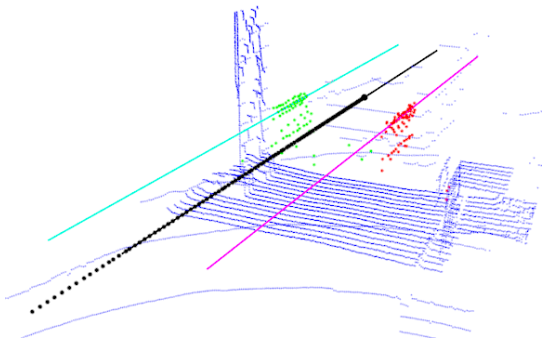

Fig. 5: Illustration of the trajectory estimation. The image shows the lines regressed for both sides of the truck (cyan and pink lines) using the set of points of the border of the bucket (green and red points respectively), as well as the average line defining the trajectory (black line).

there is a truck in the scene, therefore, the current data is stored and the measurement process continues. Otherwise, the data is discard. The threshold was defined empirically in order to account for noise in the measurements (such as dust particles that could be confused as part of the truck).

3) LiDAR Sensors Calibration: For building the final dense point cloud of the truck, the data from both LiDARs, V^i_{pos} and V^j_{vol}, need to be integrated. Therefore, this step aims at finding the relative pose between the two LiDARs. The calibration process to obtain the parameters to project points from one LiDAR coordinate system to the other is performed just once and semi-automatically. First, an initial guess for the pose value is provided manually (such as clicking in corresponding points in both point clouds). Finally, the initial guess is fine-tunned using the Generalized Iterative Closest Point algorithm (G-ICP) [8], which returns a transformation to align the two point clouds V^i_{pos} and V^j_{vol}. The output of this step is a transformation that is used to map the point cloud V^j_{vol} from the coordinate system of the top LiDAR to the road LiDAR. For simplicity, from now on, the same notation (V^j_{vol}) is used to represent the point cloud from the top LiDAR in the coordinate system of the road LiDAR. Figure 4 illustrates a pair of point clouds before and after the calibration of the sensors.

4) Truck Path Estimation: This step aims at estimating the linear trajectory performed by the truck during the measurement. The trajectory is assumed to be linear, i.e., the driver should not turn the wheel while going through the measurement. This was found to be a reasonable assumption since the driver has to pass underneath the measurement structure and it would be under measurement just for a short period. The points of the side of the truck bucket were used for regressing the line defining the truck trajectory. Basically, the data from the top LiDAR V^{j-}_{vol} was used to isolate the points from both sides of the bucket. After cleaning, V^{j-}_{vol} only comprises points belonging to the truck. Since V^{j-}_{vol} is given in spherical coordinates, the first and the last points of the data package are actually the points from both side of the bucket (first and last points that hit the truck in one LiDAR revolution). The points representing each side of the bucket are isolated from the others in two groups that are used for estimating the two lines, left L_{ltrj} and right L_{rtrj} sides, (through a Linear Regression algorithm and with RANSAC to reduce the influence of noise) that define the lateral of the truck bucket. In order to avoid influence from the frontal part of the truck and the bucket (which is not linear or not parallel), the point clouds from the end of the bucket are selected for regressing the lines. The points used for the regression are collected from the point clouds V^{j-}_{vol} in which less than a half of the lasers (8 out of 16 rays) hit the truck, i.e., when only the end of the truck bucket is visible. Finally, the central trajectory line L_{trj} is estimated by averaging the points and the vectors defining L_{ltrj} and L_{rtrj} (assuming the line equation in R^3 and described by a point and a vector). Figure 5 illustrates the process of trajectory estimation. The output of this step is a linear trajectory (described by the line L_{trj}) used to restrict the possible positions of the truck along a line.

5) Truck Position Estimation: This step aims at estimating the position of the truck along the trajectory line L_{trj} at each timestamp. Since the trajectory line is perpendicular to the back of the truck bucket and the data from the road LiDAR V^{j-}_{pos} hits both, the back and the lateral of the bucket, the simple projection of V^{j-}_{pos} in L_{trj} should accumulate a high number of points in the back of the bucket. Therefore, the point with more neighbors would represent the back of the bucket and could be used as position. The points obtained by projecting V^{j-}_{pos} into L_{trj} are defined as V^{j-}_{trj}. The truck position p^j_{trk} (back of the bucket) is the point from V^{j-}_{trj} with the highest number of neighbors within a region of 30cm radius. The point is obtained by creating a KD-Tree with the projected points V^{j-}_{trj} and counting the number of neighbors within the defined radius. Figure 6 left illustrates the position estimation process.

The projection of a point p into a line $L = p_l + v_l t$ in R^3, where p_l is a point, v_l is a unit vector and t is a scalar, is given by equation 1:

$$\begin{aligned} p_{prj} &= p_l + v_l t_p \\ t_p &= v_l.(p - p_l) \end{aligned} \tag{1}$$

The equation 1 is used for projecting the points of V^{j-}_{pos} onto L_{trj}.

6) Point Cloud Coordinate System Alignment: This step aims at mapping the point clouds from every timestamp V^{i-}_{vol}

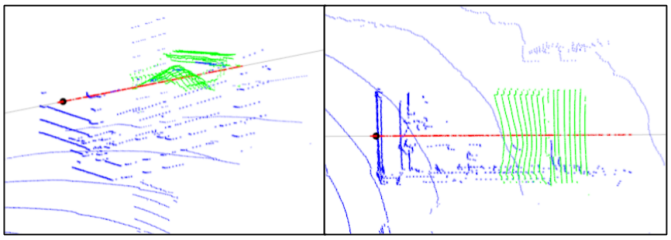

Fig. 6: Illustration of the position estimation process. This figure shows a 3D view (left) and top view (right) of the points V^i_{pos} (blue points) being projected in the trajectory line L_{trj} (red points). The black point represents the selected projected point with higher number of neighbors.

and V^{j-}_{pos} to a same coordinate system at timestamp 0. The point clouds are mapped using a transformation T^b_a that maps a set of point from the coordinate system of the timestamp a to the coordinate system of the timestamp b. The transformation is created using the points p^j_{trk} of different timestamps, as defined in equation 2:

$$T^b_a = \begin{bmatrix} I & (p^b_{trk} - p^a_{trk}) \\ 0 & 1 \end{bmatrix} \tag{2}$$

With the transformations defined, the point clouds can be mapped to the reference coordinate system at timestamp 0 using the equation 3:

$$V^{0-}_{vol} = T^0_i V^{i-}_{vol}; V^{0-}_{pos} = T^0_j V^{j-}_{pos} \tag{3}$$

Since data collection from both sensors are not synchronized, data coming from different sensors might not share the same timestamp. Therefore, the truck position p^j_{trk} of the point cloud of the road LiDAR V^{j-}_{pos}, as calculated in section II-A5, cannot be directly used for the point cloud of the top LiDAR V^{i-}_{vol}. In order to estimate the p^i_{trk} for the point cloud V^{i-}_{vol}, an interpolation is performed between p^j_{trk} and p^{j+1}_{trk}, where j and $j+1$ are respectively the closest timestamp before and after the timestamp i. The point interpolation is defined in equation 4:

$$p^i_{trk} = p^j_{trk}(1 - \Delta t^i_j / \Delta^{j+1}_j) + p^{j+1}_{trk}(\Delta t^{j+1}_i / \Delta^{j+1}_j) \tag{4}$$

where, Δt^b_a is the time difference between the timestamps a and b.

Once all point clouds are mapped to the same coordinate system at timestamp 0, they are grouped in a single dense point cloud V_{trk} representing the reconstructed truck. The output of this step is a dense point cloud V_{trk} including the points from every timestamp of the LiDAR measurements.

B. Load Reconstruction

This module receives a dense point cloud V_{trk} from the previous module, as well as a model of the empty bucket, and creates a mesh representing the load inside the truck bucket. The model has two parts that are aligned in the same coordinate system, a point cloud representing the external part and a mesh representing the internal part. The whole process comprises the following steps: bucket alignment, non-load points removal, bucket base creation and mesh reconstruction.

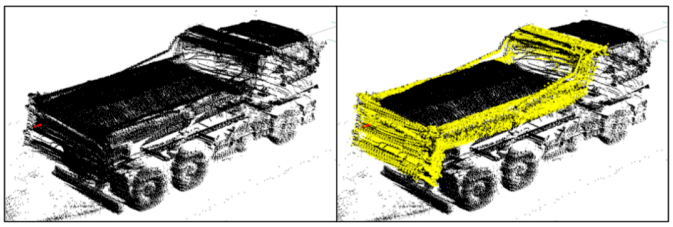

Fig. 7: The left image illustrates the dense point cloud V_{trk} representing the reconstructed truck. The point cloud uses LiDAR measurements from all timestamps. The right image illustrates the result of the alignment between the truck bucket model (yellow points) and the dense point cloud (black points).

1) Bucket Alignment: This step aims aligning the empty bucket model (given as input for the method) with the dense point cloud of the truck in order to allow defining the set of points that belong to the load. The alignment is performed automatically using the Generalized ICP (G-ICP) algorithm. The initial pose guess of the model for the G-ICP is calculated using the edges from the side of the bucket. A coordinate system is created using the middle point on the edge of the back of the bucket as origin (to map to p^0_{trk}), a vector pointing to the front of the bucket as x (to map to v_l from L_{trj}), a point on the right side of the bucket edge to generate the other axes (to map to the axes generated by a point along L_{rtrj}). Subsequently, the model is aligned to the point cloud using the defined coordinate systems so that the G-ICP can fine tune the alignment. Figure 9 (second column) illustrates the result of the alignment of the truck bucket model with the dense point cloud. The output of this step is the empty bucket model (the point cloud V_{bkt} and the mesh M_{bkt}) aligned with the truck in the same coordinate system of the dense point cloud V_{trk}.

2) Non-Load Points Removal: This step aims at removing points that are not representing the load inside the bucket. Since the models of the empty bucket (point cloud V_{bkt} and mesh M_{bkt}) are already aligned with the dense point cloud, they can be used to choose the points that belong to the load. The first part of the removal is performed with the aid of the simplified mesh M_{bkt}. The mesh M_{bkt} was created with normals pointing outside the bucket and describing the interior of the bucket. Hence, a point of the dense point cloud V_{trk} does not belong to the load (i.e., is not inside the bucket) if it is in front of any face of the mesh M_{bkt}. Therefore, the point is removed if the dot product between the normal of any face and its vector from the load point is negative, as defined in equation 5:

$$\begin{cases} remove, & \text{if } (p_{fac} - p_{loa}) \cdot N < 0 \\ stay, & \text{otherwise} \end{cases} \tag{5}$$

where, p_{loa} is the current point being considered for removal and, p_{fac} and N are respectively a point and the normal of

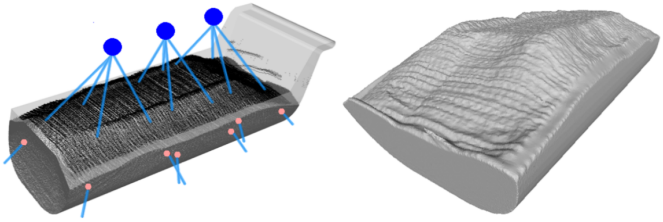

Fig. 8: Illustration of the result of the removal of external points and creation of points of the base (left), and result of the mesh reconstruction (right). The three blue dots are artificially ceated to shoot rays (blue lines) towards the load points until they hit the base (orange dots are the new base points).

a triangle face of M_{bkt}. For each point in V_{trk}, all faces of M_{bkt} are verified with equation 5.

The second part of the removal is performed with the aid of the bucket model point cloud V_{bkt}. It aims at removing noisy points that might have remained inside the bucket (such as points in the lateral of the bucket coming from a wrongly aligned point cloud). Therefore, a KD-Tree is created with the V_{bkt} so that points from V_{trk} can be verified and removed if they are too close (less than 10cm) to any point of V_{bkt}. The output of this step is the dense point cloud V^{-}_{trk} containing only the points inside the bucket. Figure 8 illustrates the result of the removal of the points outside the bucket.

3) Bucket Base Creation: This step aims at creating the set of points representing base of the load that touches the bucket. The base was created by shooting rays from the top of the bucket (one close to the end, one close to the middle and one close to the beginning) and passing through the load points (see Figure 8). Each of these three shooting points shoots rays towards a set of points of the load V^{-}_{trk} (each point of the load belongs to the set of the closest shooting point). The intersection between the rays and the faces of the mesh M_{bkt} creates a new point p_{bas} for the base. The coordinates of a point p_{bas} are obtained with the Möller-Trumbore intersection algorithm [9]. This method receives as input the vertices of a triangle face and a line in R^3 and outputs the intersection point if it exists. The output of this step is a point cloud representing the points of the load V_{loa} including the point of the base that touches the bucket.

4) Mesh Reconstruction: This step aims at creating a closed mesh out of the point cloud of the load V_{loa} and is accomplished using the Poisson Surface Reconstruction algorithm [10]. The algorithm uses a Poisson formulation that considers all points at once, without resorting to heuristic spatial partitioning or blending. Such approach makes the algorithm highly resilient to noise. However, the generated mesh might still come with some imperfections like wholes, flipped edges, etc. Therefore, the method Quadric Edge Collapse Decimation (extracted from the open-source software MeshLab) is applied to correct such issues. The method is based on the Surface Simplification using Quadric Error Metrics proposed in [11]. Figure 8 right illustrate the mesh reconstruction results.

C. Volume Estimation

Finally, this module receives the reconstructed mesh M_{loa} from the previous step and outputs the measured volume. The volume is measured using an algorithm proposed by Brian Mirtich [12] that computes a series of polyhedral mass properties including volume.

III. Experimental Methodology

This section describes the experimental methodology employed for evaluating the proposed system.

A. Dataset

The proposed system was evaluated in a mining company in real conditions of operation. The experiments were performed using a single truck with three different loads of ore powder. Two loads came from a stockyard (YARD_L1, and YARD_L2) in which piles of ore powder are stored. The third one (SILO_L1) was originated from a silo. For each load, the truck passed through the measurement structure twice resulting in a total of six datasets. Each dataset contains the set of all LiDARs' point clouds captured while the truck passed through the structure. The datasets with loads of ore powder from the stockyard will be referred to as YARD_L1M1, YARD_L1M2, YARD_L2M1, YARD_L2M2. The suffix LiMj is used to represent the j-th measurement of the i-th load. Likewise, the remaining dataset with loads of ore powder from the silo will be referred as SILO_L1M1 and SILO_L1M2. It is worth mentioning that the process of collecting the datasets was challenging since the company had to concede a truck, a driver, and an employee that were departed from the company's important activities during the data collection.

The ground truth weights of the loads were measured by weighting the loaded trucks using a scale from the company and subtracting the weight of the empty truck. The weights of YARD_L1, YARD_L2, and SILO_L1 are 15.920t, 16.260t, and 15.500t, respectively. The volume of the loads are derived from the weights. The company informed that the average density of the ore powder is 2.3t/m^3. The ore powder is subject to different processes in the factory that can change its density. In the stockyard, for instance, the piles of powder are watered to reduce the emission of solid particles to the atmosphere.

B. Metrics

The absolute error (AE) is used for evaluating the accuracy of the values obtained using the proposed system. This metric is given by the absolute difference between the estimates values and ground-truth values. To summarize sets of sample absolute errors, the mean absolute error (MAE) is used. The computational performance of the proposed system is evaluated by measuring the average execution time of the modules as well as the average total execution time (these averages are taken over datasets).

C. Setup

The modules of the software were implemented in C/C++ using a in house version of carmen[1]. The implementations of

[1]https://github.com/LCAD-UFES/carmen_lcad/

G-ICP and the Poisson surface reconstruction [10] are from the Point Cloud Library (PCL). The system was executed in a notebook with processor Intel Core i7-4600M 2.90GHz and 8GB of RAM. The parameters used in the algorithms are the following. The number of iterations and maximum correspondence distance of the G-ICP are set to 2000 iterations and 0.5m, respectively. Finally, the depth value of the Poisson Mesh Reconstruction algorithm is set to 10. All of these parameters are chosen empirically.

D. Experiments

A set of four experiments were performed using real data from a factory followed by qualitative and quantitative analysis. Each experiment is explained in details below.

1) Consistency Evaluation: The volumes estimated by the system in different measurements of the same load are compared. The same volumes should be returned by the system given that the loads are the same. This experiment aims at measuring the errors in the inner processes of the system.

2) Weight Accuracy: Using the average density of the ore powder provided by the company and the volumes estimated by the system, the weights of the loads are estimated. The MAE and the mean percentual error between the estimated and ground-truth (using the company's scale) weights is used for evaluating the system's accuracy.

3) Accuracy Cross-Loads: It is worth noting that the average ore powder density may not be accurate due to the ways the ore powder is processed in the company (see section III-A). Therefore, the comparison across load measurements for the same material may provided a more realistic ground truth. In this experiment, one load is selected for computing the ore powder density, and another for evaluating the system accuracy. Since only the datasets from the stockyard have measurements for more than one load and they have approximately the same density, the silo data is not used in this experiment. The system is used to estimate the volume in both measurements of a load. Using these volumes and the load weight, two sample densities are computed. The final ore powder density is obtained by averaging the sample densities. The ground-truth volume of the other load is obtained by dividing its weight by the computed density. The ground-truth is then compared with the volumes estimated by the system in the two remaining measurements. Accuracy is also measured in inverting the loads for density estimation.

4) Accuracy Cross-Measurements: This experiment is similar to the previous one, however one measurement from each load is selected and used for computing the density and the other measurement is used for evaluating the system. The estimate of the ore powder density is also obtained by averaging sample densities computed using the loads' weights and the volumes estimated by system. Accuracy is also measured in inverting the loads for density estimation.

5) Efficiency Evaluation: The goal of the proposed system is to increase the efficiency of the company in the process of estimating the amount of goods being transported by trucks. The time necessary for computing a volume given sensors' data is averaged across datasets. The average execution time is compared with the time necessary for weighting the truck under the assumption that both processes aim at measuring the amount of materials being transported by the truck.

IV. Results and Discussion

The results for the qualitative evaluation of the system are shown in Figure 9. The quality of the meshes matches the pictures and indicate that the system's modules succeeded in real world operation.

The volumes estimated for each of the datasets are presented in table I, along with load weights computed using the average ore powder density (2.3 Kg / m^3) provided by the company, the ground-truth weight, the absolute error, and the percentual error. The consistency evaluation shows that the absolute difference between the volumes estimated for the same loads is 0.1m^3. Choosing the second measurement as the reference, the average percentual error is 1.48%. The system achieved a mean absolute error of 0.69t and a mean percentual error of 4.41%. The high mean percentual error is mainly due to incorrect (given the imprecise density) weight estimates for the datasets SILO_L1M1 and SILO_L1M2. Considering only the datasets with loads from the stockyard, the mean absolute error and the mean percentual error reduce to 0.29t and 1.84%, respectively. It is worth restating that in the silo, water is added to the ore powder which can change the powder density.

TABLE I: Evaluation of the load weights computed using the volumes estimated by the system.

Dataset	$V_{est}(m^3)$	$W_{est}(t)$	$W_{gt}(t)$	AE(t)	% Error
YARD_L1M1	6.76	15.56	15.92	0.36	2.27
YARD_L1M2	6.68	15.36	15.92	0.56	3.52
YARD_L2M1	7.14	16.41	16.26	0.15	0.94
YARD_L2M2	7.03	16.16	16.26	0.10	0.61
SILO_L1M1	7.44	17.12	15.50	1.62	10.43
SILO_L1M2	7.33	16.85	15.50	1.35	8.70
			Average	0.69	4.41

Using the cross-load approach, the volumes obtained for the datasets YARD_L1 and YARD_L1 are 6.93m^3 and 6.86m^3, respectively. The MAE of the volumes estimated by system when compared to these values is 0.21m^3 which gives a mean percentual error of 3.1%. Similarly, using the cross-measurements approach, the volumes obtained for the datasets are 6.78m^3 for YARD_L1M1, 6.87m^3 for YARD_L1M2, 6.92m^3 for YARD_L2M1, and 7.02 for YARD_L2M2. The MAE of the system when compared to these values is 0.11m^3 which gives a mean percentual error of 1.55%.

Estimating the volumes from the sensors' data took on average 88.9s. The truck reconstruction step and the load reconstruction step took on average 0.56s, and 88.35s, respectively. Since weighting each load in the data collection day took at least an hour. Assuming that this long delay for using the scales is common, we conclude that the proposed system is more efficient than the current process used in the factory for quantifying transported materials.

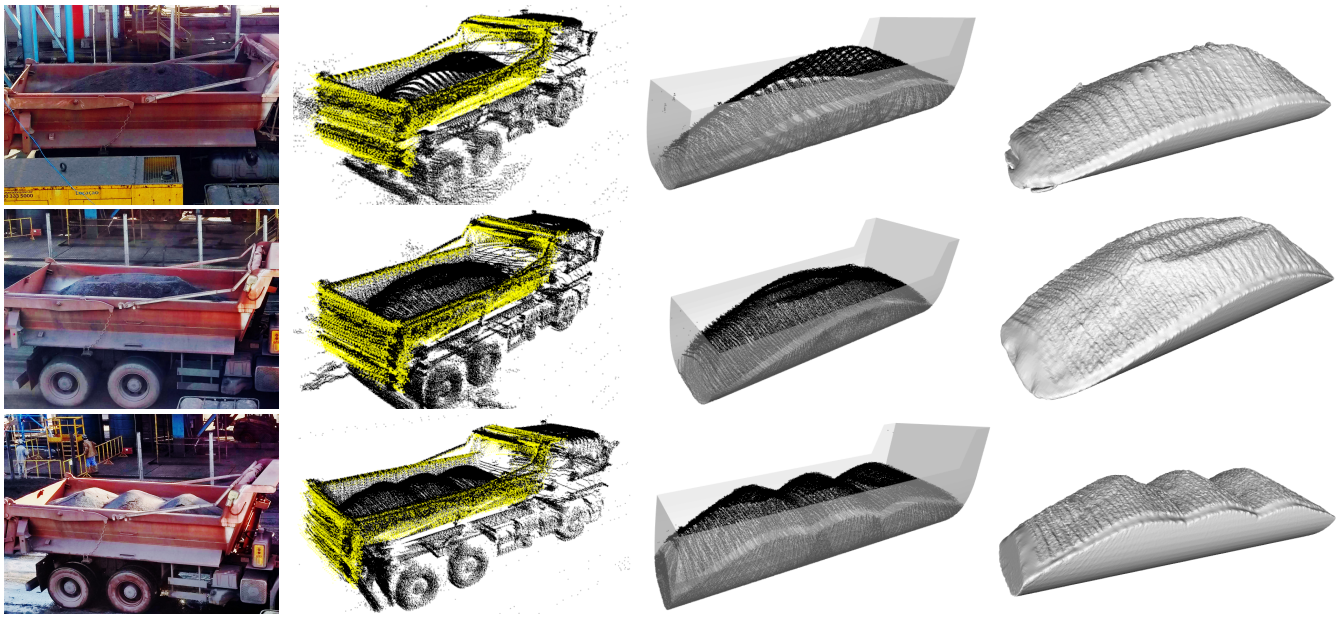

Fig. 9: Qualitative analysis of the reconstruction of the truck and of the truckload using the proposed system. The rows represent measurements of different loads for the datasets YARD_L1M1 (first row), YARD_L2M1 (second row), and SILO_L1M1 (third row). The column presents from left to right: truckload pictures, reconstruction of the truck with alignment of the bucket model, reconstruction of the load bottom using the inner part of the truck bucket model and the meshes built from the truckload.

V. Conclusion

This work proposed a new technique for estimating the volume of loads from moving trucks using two LiDARs. The LiDARs' data are used for reconstructing the truck and its load. With the LiDAR measurements of the truck, a mesh of the load is created with the aid of an *a priori* model of the truck bucket. The load volume is finally estimated from the mesh. The system was deployed and evaluated in a mining company in real conditions of operation. Experimental results showed that the system can be successfully used for reconstructing a truck and loads of ore powder. Using the average density of the ore powder provided by the company, the weight of the loads could be estimated with an mean absolute error of 0.69t which is equivalent to 4.41% of the real weight on average. The ore powder used in the experiments come from different storage units (a stockyard and a silo). The processes used in these units can affect the ore powder density. Considering only the loads from the stockyard, the mean absolute error and mean percentual errors reduce to 0.29t and 1.84%, respectively. Future work includes more experiments in real scenarios and with ground truths for the volume.

Acknowledgment

Financed in part by Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq, Brazil, grants 311120/2016-4 and 311504/2017-5); by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brazil (CAPES) - Finance Code 001; and by Vale/FAPES (grant 75537958/16).

References

[1] B. A. B. Correia, R. Davies, F. D. Carvalho, and F. C. Rodrigues, "Computer vision system for the automatic measurement of volumes of wood," vol. 1989, 1993. [Online]. Available: https://doi.org/10.1117/12.164869

[2] H. Marshall *et al.*, "Automated log counting: proof of concept algorithm." in *Proceedings of SilviLaser 2011, 11th International Conference on LiDAR Applications for Assessing Forest Ecosystems, University of Tasmania, Australia, 16-20 October 2011*. Conference Secretariat, 2011, pp. 1–13.

[3] M. Acuna and A. Sosa, "Automated volumetric measurements of truckloads through multi-view photogrammetry and 3d reconstruction software," *Croatian Journal of Forest Engineering: Journal for Theory and Application of Forestry Engineering*, vol. 40, no. 1, pp. 151–162, 2019.

[4] H. Anwar, S. M. Abbas, A. Muhammad, and K. Berns, "Volumetric estimation of contained soil using 3d sensors," in *Intl. Commercial Vehicle Technology Symposium*, 2014, pp. 11–13.

[5] E. Duff, "Automated volume estimation of haul-truck loads," in *Proceedings of the Australian Conference on Robotics and Automation*. CSIRO, 2000, pp. 179–184.

[6] M. Nylinder, T. Kubénka, and M. Hultnäs, "Roundwood measurement of truck loads by laser scanning," *Field study at Arauco pulp mill Nueva Aldea*, pp. 1–9, 2008.

[7] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM*, vol. 18, no. 9, pp. 509–517, Sep. 1975. [Online]. Available: http://doi.acm.org/10.1145/361002.361007

[8] A. Segal, D. Haehnel, and S. Thrun, "Generalized-icp." in *Robotics: science and systems*, vol. 2, no. 4, 2009, p. 435.

[9] T. Möller and B. Trumbore, "Fast, minimum storage ray/triangle intersection," in *ACM SIGGRAPH 2005 Courses*. ACM, 2005, p. 7.

[10] M. Kazhdan, M. Bolitho, and H. Hoppe, "Poisson surface reconstruction," in *Proceedings of the fourth Eurographics symposium on Geometry processing*, vol. 7, 2006.

[11] M. Garland and P. S. Heckbert, "Surface simplification using quadric error metrics," in *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., 1997, pp. 209–216.

[12] B. Mirtich, "Fast and accurate computation of polyhedral mass properties," *Journal of graphics tools*, vol. 1, no. 2, pp. 31–50, 1996.