Evaluation of sidewalks based on 3D images

Jose Barros, IST Manuel Marques, IST and Joao Costeira, IST

Abstract—Walkability is defined, generally, as to which extent an urban environment is considered prone to walking. Having different tools in order to measure this index, helps the experts to better address the problems that contribute for a lower walkability index. This work presents a solution for the assessment of pedestrian infrastructures based on 3D scanning, using 3D cameras, and scene segmentation, addressing the contribution that the quality of sidewalks influence walkability. The methodology applied returns a classification of each sidewalk analyzed, helping the professionals to understand where the problems reside and address them. It was applied in an area around Instituto Superior Tcnico in Lisbon, Portugal, evaluating the sidewalks in the area and, potentially, generating data that can be used in order to improve walkability. The results show that, in most cases, the method is accurate enough to match reality despite some cases where the pre-processing was not successful enough, and thus, providing inaccurate assessments. It is important to show these cases in order to emphasize that the method is mostly reliable, but, in some cases, and since we are working with real data, it is not flawless.

Index Terms—Walkability, Scene segmentation, 3D cameras, Sidewalk assessment, Sidewalk quality.

1 Introduction

Walkability has become a very popular subject nowadays. It is associated with a healthy lifestyle and with environmental achievements. In an overall statement it is defined as how easy it is for people to walk around their areas for any reason, like work, shopping or physical activity. So, it has become one of the top shelf priorities to build cities whose environment has subjective and objective characteristics that might urge people to walk in their day to day. If people are more willing to walk, it can lead to a reduced energy consumption and help building a better health [1].

Green areas [2], the proximity and connectivity [3], aesthetics [4] and the urban design [5] are some of the factors that define how walkable an area can be. The green areas are associated with a healthier region and therefore more appealing to people. How close the points of interest are to one another is also important as most people are not willing to walk big distances to satisfy their needs. The connectivity study aims to see how many routes are available to go from a point to another and how easy it is to find those routes. Aesthetics and urban design are related on the appearance part of the building, the more appealing the space is, the more people will enjoy it.

As sidewalks are the most common surface used for people to move around according to their needs, they are a big influencing factor on a citys walkability index [6]. Their quality in terms of construction is of the most relevance for a study on walkability since people may choose not to walk simply because there isnt an available and safe sidewalk that connects them to their destination or, even if they exist, the state theyre in, might be so degraded that it makes it impossible to walk on. In this paper we came up with a system that is able to analyze a set of data, which are images taken with a 3D camera. After the images are taken we get the 3D reconstruction of the area, and more specifically, from the sidewalk, using a simultaneous localization and mapping system and then we can determine what objects are present on the sidewalk, and we calculate the best path which is the wider one. As this is done we also built a sidewalk descriptor that will analyze the surface of the sidewalk. Also, in order to get the classification and since the descriptor is basically a histogram that characterizes the sidewalk, we used an optimization problem called Earth Mover's Distance [7] and, with the distance between the histograms we intended to classify and a priorly obtained descriptor of the best possible sidewalk that we considered, we would determine the class (Good, Medium or Bad) the sidewalk would belong based on how far its descriptor was from the optimal one. We hope that the results provide us accurate descriptors that can help the city council to repair and provide better infrastructures to improve citie's walkability.

2 State of the art

During our research it became clear that we could separate the problem into two strands. One would be understanding the concept of walkability and what it represents, and second, the understanding the 3D environment.

2.1 Walkability

While studying walkability we found out that in past researches around the world its proven that the quality, appealing and existence of physical infrastructures are of the most importance to make any place a walkable one. How appealing is a certain place is determined by multiple factors that help us to establish a rating for each region we study. For example, the abundance of green zones, or the lack of it, its most definitely relevant for a pleasant walk around those regions as it is associated with a healthier lifestyle [1].

The design of a region is also relevant for walkability since it gives people a more appealing environment to walk on. Theres a study [5] that establishes five concepts that are used to assess the design of a certain street/venue. They start with imageability which describes how a place is seen, i.e., its trademark image. Then they address visual enclosure

[3]. Human scale defines how the dimensions of the physical elements are matchable to the human size and it is also associated with how fast can a human walk between points of interest in that same street. Transparency is related with what people can see beyond the street buildings and corner. And finally, they address complexity, which describes how rich, visually speaking, the street is.

2.2 Understanding the 3D Environment

The existence of infrastructures that allow a person to walk between his/her destinations is evaluated by the connectivity index of a region. How easy and fast it is to walk from point A to point B as well as how many alternative routes there are, are main features of a walkable environment [4]. Also, the dwelling density, linkage between different neighborhoods, land use and retail network combined can determine the walkability of a certain city [8].

Quality of the infrastructures is most definitely one of the highest contributing factors that motivate people to walk around their city on their day to day. Namely the quality of the sidewalks, i.e., how well built they are, if they have enough width for people to walk on, the presence of obstacles and how flat they are. Being able to help the city council to properly maintain this infrastructure is what moves us towards this specific problem on walkability. There have been similar work on this field, for example, the use of an Android app that would scan the sidewalks and evaluate them and inform if there are any repairs needed [6]. They focused their testing on seeing how accessible the streets are for a wheelchair user, we intend to have a more general approach and being able to give useful information to everyone.

3 Proposed Solution

To implement our solution, we designed a device, a box, that is set to scan and analyze the sidewalks of any location. Since the purpose of this device is to scan while walking, to be portable if of the most importance. After the scanning, some of the processing is done online and the rest is offline. The offline part of the work may be done inside the box or outside it, since it provides the outputs necessary to use the data outside of its own environment.

Next we provide the list of every component that's inside the box and that we need to have it successfully functioning. We have 3D camera, Asus Xtion Live Pro camera, a LCD 7 Touch Screen, an Up Board, a GPS Module, a battery and a step-down converter.

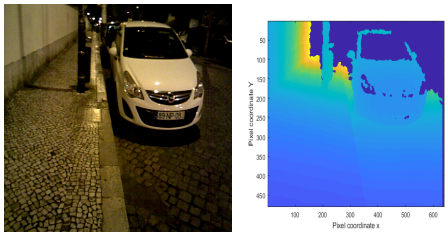
For our work, the processing inside the the box, more specifically, in the Up Board, takes some computation power to come up with the scanning and assessment in reasonable time. And, as such, the processor's capacity is something to consider. The touch screen is also important, it will allow the user to take full advantage of the solution and shows the information needed to help the user throughout the process. The camera, of course, is the most important component as it allows the scanning to happen to begin with. The information we are able to gather from the camera is essential for the rest of the work.

For the device to be portable we needed the battery to provide the power to all the rest of the components. For the components to fit inside the device we designed the box and a case for the battery using 3D printing.

3.1 Simultaneous localization and mapping

The first step of our solution, as stated before, is the usage of SLAM, more specifically, ORB-SLAM. The system that it is built on has three main threads running at the same time. One is the tracking to localize the camera with each of the frames available. The tracking is done by, firstly, taking the pair RGB-Depth images and, then, considers the left and right side views.

An example on the input that our system takes is shown in 1.



(a) **(b)**

Fig. 1: On image (a) we have one RGB input example and on (b) an input depth example.

When we provide the pair of images mentioned before, it and extracts features at detected salient keypoint locations. These keypoints, called stereo keypoints, are classified as close and far. On this method, the keypoints are given by the following equation

$$x_s = (u_L, v_L, u_R), \tag{1}$$

being (u_L, v_L) the coordinates on the left image and u_R the horizontal coordinate in the right image. For stereo cameras, which we do not use or need, ORB features are extracted from the left image and simultaneously matched to the right image. For RGB-D cameras, which is our case, the features are extracted from the RGB image. Then, each feature's depth value d into a virtual right coordinate

$$u_R = u_L - \frac{f_x * b}{d}, \tag{2}$$

where f_x is the horizontal focal length and b is the baseline between the structured light projector and the infrared camera. The keypoints are classified as close if its associated depth is less than 40 times the stereo/RGB-D baseline. Otherwise, it's classified as far.

This method performs Bundle Adjustment (BA) to optimize the camera pose in the tracking thread to optimize a local window of keyframes, which are windows that contain a set of keypoints, and points in the local mapping thread.

First the system performs Motion-only BA to optimize the camera orientation $\mathbf{R}$ and position $\mathbf{t}$ by minimizing the reprojection error between matched 3D points, X^i, in world coordinates and keypoints, $x^i_{(.)}$. This optimization problem is shown in 3

$$(R,t)^* = \underset{R,t}{\operatorname{argmin}} \sum_{i \in B} \rho \left(\left\| A^i - (RB^i + t) \right\|_\Sigma^2 \right) \quad \text{s. t.: } RR^T = I \tag{3}$$

where A_i and B_i are two 3D points, ρ is the robust Huber cost function and Σ is the covariance matrix associated to the scale of the keypoint and I is the identity matrix.

Local BA optimizes a set of keyframes, K_L and all points seen in those keyframes, P_L while full BA is a specific case of local BA, where all keyframes and points in the local map are optimized. In short, first, it takes every keyframe coming from the extracted ORB features and optimizes them locally as it progresses. The local BA optimization problem is defined in 4

$$(R_l, t_l)^* = \underset{R_l, t_l}{\operatorname{argmin}} \sum_{k \in K_L} \sum_{j \in X_k} \rho \left(\left\| x_k^j - (R_l X^j + t_l) \right\|_\Sigma^2 \right) \tag{4}$$

where set of matches between points in keyframes P_L and keypoints in a keyframe k is given by X_k.

Once this is done, the system proceeds to optimize the optimized set of keyframes to generate the full map. Note that, full BA is only performed after the loop closing thread, as it is a costly process and, performing it on a different thread, allows the system to keep creating the map and detecting loops.

Despite this ORB-SLAM system performing loop-closing and full BA duties, on our algorithm, it is not necessary and we don't use it since we are trying to show the user, a 3D reconstruction of the exact environment he's scanning in real-time, therefore, we only need the local process.

The relocalization is possible using this SLAM approach because it has built in place recognition, i.e., it saves previous mapped scenes so that we are able to relocate the camera in case of tracking failures. The system keeps a graph linking any two keyframes corresponding to the same point and a spanning tree that connects all the keyframes so that we can retrieve local windows of keyframes which enables the tracking and mapping to be done locally.

This system uses ORB features for tracking, mapping and place recognition as they are robust to rotation and scale. They are, also, faster to extract and match which allows the system to work in real-time.

In 2 we can see the process of tracking with a certain amount of keypoints and the 3D map creation based on the camera trajectory.

(a) (b)

Fig. 2: On image (a) we have the tracking realized on the RGB image with the detected keyframes and, on (b) the 3D map being created as the camera moves along the trajectory.

Also, we must distinguish between the online strand, where the algorithm uses batches of seven to ten images and shows the user what is happening and what is being processed, and the offline strand where the 3D map is being generated with those frames batches. With the online information the user can check whether the data is stable to commence processing which is very important in terms of getting good results. Since we are mostly interested to use ORB-SLAM in real-time, to show the user what's being scanned, we take full advantage of the local map that is created and therefore we don't need the full map to be created saving computational costs.

3.2 Plane and object segmentation

As ORB-SLAM builds the 3D reconstruction of the scene, we analyze each frame, using the pair depth-RGB images, to determine the obstacles present on the sidewalk and calculate the optimal path that the pedestrian should take for a safer and guaranteed passage.

Estimating the normal vector and removing the sidewalk plane

First, we start by getting the 3D coordinates of every point on the frame using its depth image. We ask the user to point us where the sidewalk is, selecting 4 random points. This process of asking the user to point the system to the sidewalk only happens when the camera pose changes significantly and, to avoid selecting points outside the sidewalk with those coordinates, we ask the user to point us the sidewalk again. An example of 3 consecutive images where the algorithm didn't ask for a new set of points is shown in 3.

(a) (b)

(c)

Fig. 3: Sequence of 3 consecutive images, where the camera pose wasn't changed significantly in order to cause a mismatch from the previous clicked points and the new frame.

As we can see in 3, the sidewalk remains in the same position on all of the 3 cases, i.e., the pixels pointed out on image (a) would still be valid through images (b) and (c).

On the following pair of images, the opposite happened, the algorithm asked the user for a new set of points.

(a) **(b)**

Fig. 4: Sequence of 2 consecutive images where the algorithm required a new set of points.

We can perfectly see that, on image (a), if the user selected points on the left side of the sidewalk, when the next frame came, image (b), they would no longer be valid as they could coincide with the light pole. Now, to determine if the camera pose changes significantly we estimate the normal vector in every frame t and compare with the previous frame $t-1$, as shown in the following expression,

$$cos(\theta_t) = \frac{n_{(t-1)} \cdot n_{(t)}}{||n_{(t-1)}|| * ||n_{(t)}||}. \tag{5}$$

If both normal vectors have the same direction, the cosine will be equal to one. So, if it lowers significantly, we make the algorithm ask for new points.

Now, with these 4 points, we build a perimeter connecting all of the selected points. Inside this perimeter there is a small set of points that we gather as possible points belonging to the sidewalk. Then, we align the pixels of the depth image with the RGB image. This is done so that when we select the points on the RGB image it corresponds to that point on the depth image so that we have it easier to access the 3D coordinates of said points.To get the coordinates of each point we must have the cameras parameters (intrinsic and extrinsic), then, for the depth coordinate, we take the depth information of each pixel and convert it to meters, as the following expression shows,

$$z_i = \alpha d_i. \tag{6}$$

We proceed to get the other two coordinates as shown on equations (7) and (8).

$$x_i = \frac{z_i}{K_x} * (u_i - C_x), \tag{7}$$

$$y_i = \frac{z_i}{K_y} * (v_i - C_y), \tag{8}$$

where K_x and K_y represent the focal length in terms of pixels and u_i and v_i are the depth coordinates. C_x and C_y are the camera principal point. A 3D point is given by

$$p_i = \begin{bmatrix} x_i & y_i & z_i \end{bmatrix}^T. \tag{9}$$

From the educated guess asked to the user by our system, we have a set of N possible points belonging to the sidewalk with their respective 3D coordinates, as the following expression shows,

$$P_{sw} = \begin{bmatrix} p_1 & p_2 & \cdots & p_N \end{bmatrix}, \tag{10}$$

.

Having this small set of points, we sample them with the RANSAC algorithm to get the best set of points that fit our model. The RANSAC routine receives as input a set of points that we gathered from the user interaction, P_{sw}, the whole set of points from the frame the is being analyzed, P_s, and a threshold value.

For each RANSAC iteration we start by grabbing 20 random points to constitute our subset. We center them, P_c, by removing the centroid p_c, as shown in the following expressions,

$$p_c = \frac{1}{N} P_{sw} \mathbf{1}, \tag{11}$$

$$P_c = P_{sw} - p_c * \mathbf{1}^T, \tag{12}$$

where $\mathbf{1}$ is a column vector of ones.

After having the centered data, we can performed a singular value decomposition (SVD) to obtain the normal to the plane. Now, the singular value decomposition is the factorization of a matrix, in our case, P_c, into three different matrices of the form presented in the following expression,

$$P_c = UDV^T, \tag{13}$$

where U is the matrix, containing the left singular vectors, D is a diagonal matrix containing all the singular values of P_c and V is the matrix containing the right singular vectors transposed. Since we are applying the SVD to the points matrix, which is Nx3, with N being the total number of points, U is a 3x3 matrix and D will contain three single values in its diagonal. On our model, the plane is 2D. We confirmed it since the third single value on D is close to zero. Therefore, U, in the first two columns will have the vectors where the points are mostly represented and, the third column, will be an orthogonal vector to the plane, i.e., it will be the normal to the plane that we are looking for in order to proceed.

After the normal vector is estimated, we project all the points, P_s, onto the plane by applying dot product between the vector that contains the points and the normal vector as shown in equation (14).

$$\mathbf{p} = P_s \cdot n \tag{14}$$

where proj is the resulting array containing the projection error of each point towards the model, i.e., the sidewalk plane. From this array we collect the indexes of every value that is lower than the threshold we required when using the routine. This threshold can be changed, if we use a higher value we reduce the accuracy of the results since we are considering points with higher error, and therefore, points that might not belong to the sidewalk. On the other hand, if we use a lower value, we increase the accuracy since the error we are considering is lower, so, the chances of these indexes corresponding to points on the sidewalk are higher. These are two different approaches that we considered and tested below.

Once we know which points belong on the sidewalk we rebuild the binary image without the sidewalk and leave the remaining features of the scene to analyze. This process is illustrated in 5.

(a) (b)

Fig. 5: On image (a) we have the RGB image of the processed frame and, on image (b) the binary image with the removed floor marked by the red boundary.

As we can see, in red, were the points removed as they were considered to be in the sidewalk. The red area on the right was not supposed to be removed as it belonged to the road. This happened because the threshold isn't enough to segment properly the image, leaving some unwanted features, unfiltered.

3.2.1 Connected components and path estimation

Having the scene without the sidewalk we can apply connected components, on the binary image, to determine how many objects are present on the remaining scene. We considered only relevant objects, i.e., labels whose set was bigger than 3500 pixels.

The generated labeled image can be seen in 6.

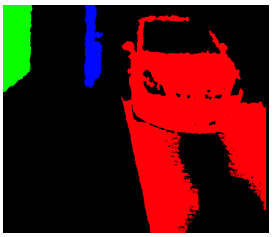


Fig. 6: Labeled image resulting from applying connected components.

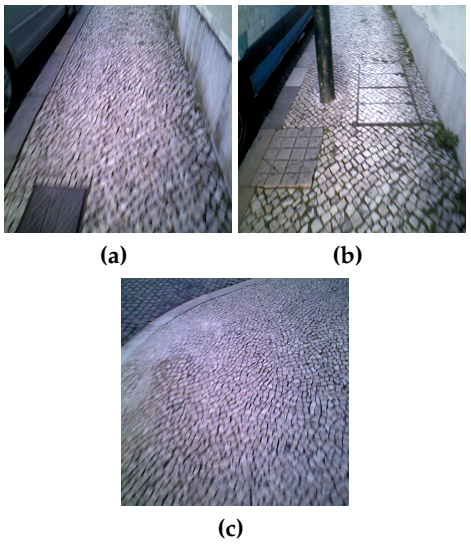
With the remaining image labeled we determine how many labels, i.e., objects, are present. Depending on how many labels were detected and validated as relevant on the scene, we would consider multiple scenarios that are listed below.

1) A wall on the left and cars on the right, i.e., objects on the left and right of the camera.
2) Apart from the side objects, we might have obstacles, like light poles or trash cans.
3) No objects around the sidewalk or obstacles in the middle.

On 7 below, we give an example for each of the scenarios.

For each of the cases we use an if-then system that considers how many labels there are and computes the walking distance (width in meters) in conformity.

For scenario 3, where theres no objects on and/or around the sidewalk, and for scenario 1, the system has it easier to determine the walking distance since it will be the full sidewalk. When we have obstacles, usually the number of labels is higher than two.



(a) (b) (c)

Fig. 7: Image (a) represents scenario 1, image (b) represents scenario 2 and image (c) represents scenario 3.

On this scenario, scenario 2, we must compute the distance on the right of the obstacle and the left distance as well so we can decide which path is optimal for the pedestrians, i.e., wider. To do this we perform k-nearest neighbors algorithm and then, comparing the two distances, we decide to take left or right depending on which distance is higher.

3.3 Sidewalk descriptor

3.3.1 Data processing

The idea of the descriptor is to have a unique feature that characterizes each portion of sidewalk. As it will be a common feature to every bit of sidewalk that we analyze, we can compare one another in terms of flatness and irregularities.

To implement the descriptor, we use the removed points of the sidewalk that we computed in the plane segmentation routine and we build a binary image only with the sidewalk. In order to treat the image, i.e., to separate the sidewalk from the road or any other feature that doesn't belong and wasn't filtered by the pre-processing, we used morphological filters.

Now, there are a few available. There's Erosion, Dilation, Opening and Closing filters. The erosion filter, erodes the boundaries of the object in the image while dilation does the opposite. Opening is a combination of the two filters mentioned before, it is an erosion followed by a dilation and works very well when removing noise from images which is, at this point, the exact operation that we need in order to get the sidewalk cleared of unfiltered features. Closing is the exact opposite of the opening filter, applies a dilation followed by an erosion. This last filter we use it on the sidewalks in order to fill in the holes that are left in them by the applied threshold from removing the floor of the scene.

With the sidewalk completely separated from the irrelevant features we can, again, apply connected components and remove these features by removing all the pixels whose labels are lower than 3500. At the end of this process we

have the sidewalk cleared of bad information and ready to be analyzed. We show this process in 8.

(a) (b)

Fig. 8: Image (a) shows the sidewalk unfiltered and with some unwanted features, image (b) represents the sidewalk filtered with above filters and those features removed y connected components.

As we can see from 8 (a) to 8 (b), the features were removed and some of the holes present on (a) are filled as intended, leaving the sidewalk data ready for further processing.

3.3.2 Descriptor implementation

Since we now have the sidewalk points well cleared and ready to be analyzed we went for the simplest solution. Which was using the SVD again, since we still have the same model were looking for, the sidewalk, but improved as it is cleared of unfiltered features.

Before we apply the SVD, we use the RANSAC algorithm to choose, from the set of points, the best set of ten points that grants us with the lowest projection error array possible using the same RANSAC routine as before. The big difference from before is that, as input, the points provided are now only the ones that compose the sidewalk. This results in an array with the projection error, just like before, but this time the error represents the difference, in height, of each point to the sidewalk.

With this information we can build a histogram and observe how many points have error close to zero. The more points closer to zero, the flatter the sidewalk and vice-versa. If we have the data spread out on the histogram it means it is a very irregular sidewalk and needs repairing and monitoring.

To define the bins of the histogram we considered that a height difference of 15 centimeters would be enough to consider it as an irregularity (hole or slope). Points whose height is higher than this will be grouped on the corresponding bin.

3.3.3 Sidewalk classifier

After the whole process is complete we will have a histogram of each sidewalk that represents its quality in terms of irregularities and a stored array that forms the histogram, i.e., an array that contains the information, in percentage, of how many points belong to each bin.

To implement the classifier, we sorted all the histograms that we obtained from the best possible one to the worst. In this way we have a sorted list that works as training data and allows us to accurately classify each portion of sidewalk according to that sorted list. The criteria we use is based on the Earth Movers Distance [7]. Its a measure of the distance between histograms. This distance is calculated using a pre-computed matrix, d_{ij} that contains the absolute distance between each bin from the array that represents the histogram. This optimization problem's objective is to set a histogram P equal to a histogram Q by inserting or removing mass to each bin, as exemplified in 9 of histogram P.

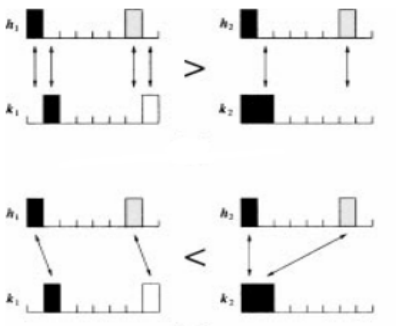


Fig. 9: The upper example represents the Euclidean distance between the 2 histograms. If each bin has 0.5 of mass in it, the distance between them would be 2, in the case on the left, and 1 on the right. Below, it's an example of the EMD calculated distance of each bin with all other bins, and, as we can see, there are less operations to make and therefore the cost is lowered and optimized. [7]

The optimization problem is defined in 15

$$
\begin{aligned}
EMD(P,Q,F) = \quad & \min_{f_{ij}} \sum_{i=1}^{m}\sum_{j=1}^{n} d_{ij} f_{ij} \\
& \text{s. t.: } f_{ij} \geq 0,\ 1 \leq i \leq m,\ \ 1 \leq j \leq n, \\
& \sum_{j=1}^{n} f_{ij} \leq w_{p_i} \quad 1 \leq i \leq m, \\
& \sum_{i=1}^{m} f_{ij} \leq w_{q_j} \quad 1 \leq j \leq n, \\
& \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} = \min\left(\sum_{i=1}^{m} w_{p_i}, \sum_{j=1}^{n} w_{q_j}\right).
\end{aligned}
\tag{15}
$$

where f_{ij} represents the flux of data which, and since in our case the data is normalized, goes between 0 and 1.

The first constraint allows for the operations of adding or removing mass to happen only in one direction, from P to Q. Second constraint limits the amount of mass that can be sent to a bin in P, i.e., we can't add to a bin over the top which is 1. The third constraint it's the opposite, we can't remove more mass than the lower bound with is zero. The last constraint forces the algorithm to make the highest number of operations possible even if the histograms are already equal to begin with. In the latter case, the cost of any operation would be null since they were already equal. The distance between each histogram is given by the sum of the costs of every operation performed to set the histograms equal to one another.

After having the distance computed with the Earth Movers algorithm, we then classify the sidewalks as good,

bad or medium. If the distance, resulting from EMD, is lower than a certain value, the sidewalk is considered good. If, otherwise, its higher than some value, its considered bad. In between those two boundary values, its classified as medium.

The boundary values were chosen considering the lowest distance scored by all the histograms and the highest distance. We then checked the classification results and the histograms to make sure that those values were adjusted to make sure that the classifications obtained were the closest possible to reality.

3.4 Summing it up

Here we will present you with the too long to read version of the proposed solution.

1) Run ORB-SLAM with sets of seven to ten frames to check whether what is being processed.
2) Compute normal vector to sidewalk plane from a given set of points after RANSAC.
3) Project all the points from the scene onto the plane to check which belong to the sidewalk.
4) Use connected components algorithm to find the objects/obstacles on the scene and determine the path.
5) Apply morphological filters to clear the unwanted data.
6) Run RANSAC to select best set of points to characterize the sidewalk.
7) Use SVD to get the projection error of each sidewalk point, in terms of height.
8) Create histogram with the latter data.
9) Compare each histogram to training data using EMD and obtain a classification for the sidewalk.

4 Results

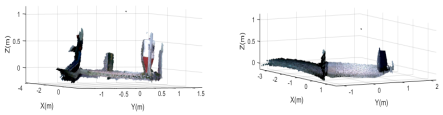
Along this section we will provide you with the various results of each part of our work with some explanations of what is going on. The data used to test our method was recorded and scanned on the streets around Instituto Superior Tcnico in Lisbon, Portugal, like shown in 10 which was generated using Google My Maps with the data generated with our GPS module.

Fig. 10: Map showing the scanned areas.

4.1 ORB-SLAM

As mentioned above, we tested the ORB-SLAM algorithm using batches of seven to ten frames. This was done so that the tracking of the points across the scene was always guaranteed.

By having the algorithm run with smaller sets of frames at a time we can show the user, in real time, the scene that is being scanned at that time. This is useful so that the user is able to see whats being processed. In the figures below well show you some examples of the various reconstructions we made using the algorithm.

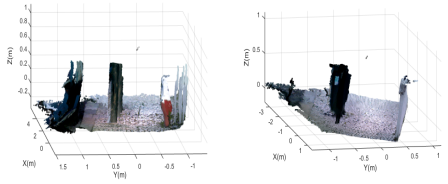


(a) **(b)**

Fig. 11: Image (a) shows the 3D reconstruction with obstacles, image (b) 3D reconstruction without obstacles.

From these two images, in 11, we can see that the algorithm works better when there are no obstacles present on the scene. The reconstruction looks much clearer in 11 (b) than in 11 (a) which has some blur and some 3D errors. This is due to the fact that the algorithm uses tracking to compute the keyframes and calculate the pose. By doing this, if the points that are being tracked belong to the sidewalk, as its color and geometry is uniform it becomes easier to track and then to reconstruct. On the obstacles case, the tracking points belonging to the obstacles will have different poses in each frame which will cause the problems that we observe in 11 (a).

Below we have two more examples of reconstructions when there are a lot of objects and obstacles present in the scene, just like in 11 (a).



(a) **(b)**

Fig. 12: Image (a) shows the 3D reconstruction with multiple objects, image (b) 3D reconstruction with a trash can in the middle of the scene.

From 12 (a) and 12 (b), and comparing with 11 (b), we can see that the less objects present on the scene, the easier the algorithm can reconstruct the scene. As there are less objects to track, the error from that tracking is significantly lower and, as such, the reconstruction becomes clearer and more accurate.

4.2 Plane and object segmentation

Since we implemented two alternatives in the segmentation algorithm, we will compare the results from both with the real measures that we took on sight.

From the binary images, that resulted from removing the floor information from the scene so that we could compute the path considering only the obstacles and the rest of the features present on the scene, we can better visualize the differences between both approaches. With a lower threshold we would obtain a cleaner scene, i.e., the data that would be further processed to obtain the path and its width, had much less outliers that could compromise the estimated measures.

We present a perfect example of the differences between the approaches in 14 (a) and 14 (b) preceded by the respective RGB image in 13.

Fig. 13: RGB image of the analyzed frame.

(a) (b)

Fig. 14: Image (a) shows the binary image with higher threshold applied, image (b) shows the binary image with lower threshold applied.

A more specific case is when we have an obstacle in the middle of the sidewalk and the decider must tell the user which path, left or right from the obstacle, is wider and, therefore, more suitable for the pedestrian. An example of this case is shown below and, as before, we present the RGB image in 15 first.

Fig. 15: RGB image for this frame.

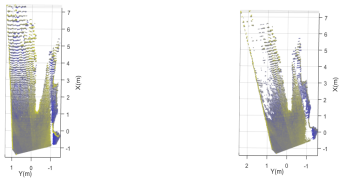
Now, we will show you the binary images of both approaches in 16.

Now the 3D point clouds for each sidewalk in 17.

On this time around, in terms of walking distance calculations, both approaches gave an accurate estimate since both thresholds were able to remove most of the points from the scene. Below, in 1, we have the distance estimates from both approaches.

(a) (b)

Fig. 16: Image (a) shows the binary image with higher threshold applied, image (b) shows the binary image with lower threshold applied.



(a) (b)

Fig. 17: Image (a) represents the 3D Point Cloud of the removed sidewalk from16 (a) in a XY view, while image (b) represents the 3D Point cloud from the removed sidewalk from 16 (b) in a XY view.

TABLE 1: Comparison between the estimates with the real value.

Real Measure(m)	1.21
Higher Threshold Estimate(m)	1.18
Lower Threshold Estimate(m)	1.19

As we said above, in terms of the walking distance both estimates are accurate, but, for the next steps of the algorithm a lower threshold data is always better because we have lower chances of getting compromising points included in the set.

4.3 Sidewalk descriptor

On this section we will present the results related to the sidewalk descriptor. Well show the histograms generated for the sidewalks analyzed and the classifier decision. For the remaining of this section we will only consider the data from the lower threshold approach.

Since we used a list of the sidewalks sorted from better to worst, we will present you first, so that we can make clearer the classifiers decision, the best sidewalks histogram possible and the worst one. These histograms are shown in 18.

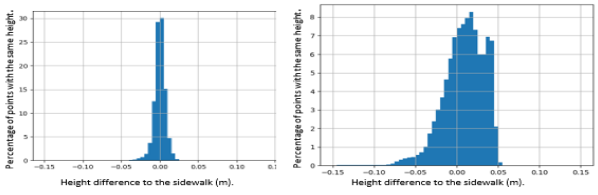


Fig. 18: On the left, the histogram representing the best possible sidewalk and on the left, the histogram representing a bad quality sidewalk.

The best possible sidewalk is the one that has no holes in it or slopes, i.e., a completely flat sidewalk. On 18, left, we have the closest representation of the best sidewalk as the highest scoring bins are the ones where the height difference is closer to zero. On the right, the sidewalk has a significant number of points with heights higher than five centimeters, which indicate that this sidewalk is in worse shape than the one on the right.

We proceed, now, to show you the 3D point cloud of each of these sidewalks mentioned above so you can clearly see the differences.

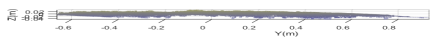


Fig. 19: 3D representation of the best sidewalk.

In 19 we can see that most of the points are grayish, i.e., most points are on the estimated sidewalk plane and, therefore, there are not significant holes or slopes.

We proceed with the worst sidewalk example below.

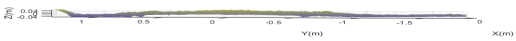


Fig. 20: 3D representation of the worst possible sidewalk.

On the sidewalk in 20, the height from the points is twice as higher than the one in Figure 4.10. And its clear the slope in the middle of the sidewalk which is non-existent in Figure 4.10. The height difference of all the points to the best sidewalk plane is the basis of our classifier, the more points there are with a higher absolute height value, the worse the sidewalk is.

We will show you below, in 2, the results of our classifier for six different sidewalks followed by their respective representative histograms and 3D point clouds so we can see how accurate our classifier was.

TABLE 2: Comparison between the estimates with the real value.

Sidewalk	Classifier's Decision
Sidewalk 1	Good
Sidewalk 2	Good
Sidewalk 5	Bad
Sidewalk 6	Bad

We will present the results below by category, i.e., we will show you the sidewalks histogram and 3D point cloud that were considered good by our classifier followed by the ones classified as medium and at last, the bad ones.

The first pair of histograms are shown in 21 below.

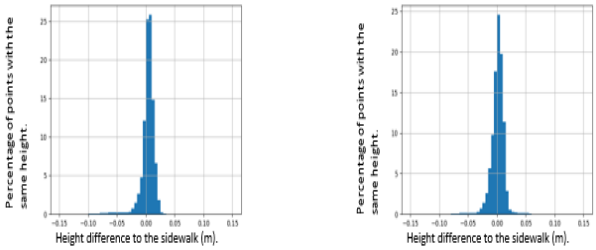


Fig. 21: On the left, the histogram representing sidewalk 1 and on the left, the histogram representing sidewalk 2.

Now, for the 3D representation we have 22 corresponding to the left histogram in 21.

Fig. 22: 3D point cloud of sidewalk 1.

For the 3D representation we have 23 corresponding to the right histogram in 21.

Fig. 23: 3D point cloud of sidewalk 2.

In comparison with the best sidewalk of 18, these two arent far off in terms of quality. We can, still, see some unfiltered data on the sidewalk of 22 that come from the wall present on the scene, but, even with these unwanted points it is considered a good sidewalk, which means that, on our data set, this might be the closest sidewalk to the best one. The sidewalk from 23 is also very close despite having, as the previous case, some unfiltered points, but this time, coming from the road.

Finally, we will show you the results of the sidewalks that were classified as bad by our classifier. On 24 we have the histograms of said sidewalks.

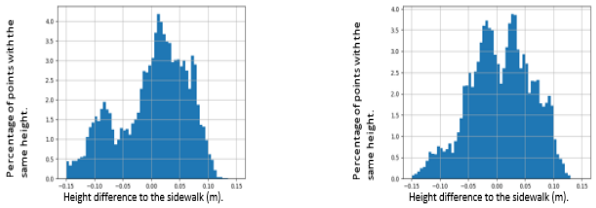


Fig. 24: On the left, the histogram representing sidewalk 5 and on the left, the histogram representing sidewalk 6.

The 3D point cloud of sidewalk 5 can be found in 25.

Fig. 25: 3D point cloud of sidewalk 5.

And below we have the 3D point cloud of sidewalk 6, in 26.

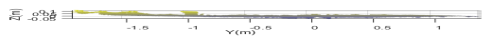


Fig. 26: 3D point cloud of sidewalk 6.

This category turned out to be a little tricky for our classifier. Since there were some images where the unfiltered points, coming either from the wall or road on the side of the sidewalk, were not completely removed as our algorithm wasnt capable of doing so using those values of threshold, they were taken into consideration when classifying. And as such, these sidewalks were considered bad. Now, on 25 theres clearly a big slope in the middle of the sidewalk which contributed massively for the low-quality index of this sidewalk. On the other hand, in 26, we can see that if it werent for the unwanted points from the road on the left, the sidewalk might have been classified into a different category. Filtering all the unwanted points is key for a more accurate classifier.

5 Conclusion

Throughout this project we implemented algorithms and techniques in order to come up with a tool that allows the user to scan, monitor and assess the quality of pedestrian infrastructures in order to improve the walkability in any location.

The work can be separated in two parts, the first, is ran in real-time, i.e., online, and the second part is done offline. On the real-time stage the user is provided with information about what is being scanned and processed so that he can decide and adjust better in order to get more accurate results. This is done by providing a 3D reconstruction of the scanned scene that allows the user o see the features that it contains and adjust accordingly. On the processing stage, offline, we get estimates for the walking path available in the scanned area that allows the user to build a reliable path for walking around. Then, finally, we have the algorithm that analyzes the sidewalk and determines its index of quality considering its state, i.e., how many holes and/or slopes are present on said sidewalk.

Sometimes, during the online presentation of the 3D reconstructions, the scene doesnt look completely clear which can compromise the users decisions and adjustments. Something to consider would be having a better algorithm to scan the frames and keep tracking of the scenes features, like an object detector. Also, on the processing algorithm, not all the sidewalks are correctly assessed and the distance estimates arent accurate enough since the segmentation isnt the most adequate. Since we use a threshold value to decide which points belong, or not, to the sidewalk, this value may not work for every situation, and, an improvement would be considering a better criterion in order to segment more accurately the scene and provide better estimates.

References

[1] S. Lee and E. Talen, "Measuring walkability: a note on auditing methods," *Journal of Urban Design*, vol. 19, no. 3, pp. 368–388, 2014.

[2] K. K. Lwin and Y. Murayama, "Modelling of urban green space walkability: Eco-friendly walk score calculator," *Computers, Environment and Urban Systems*, vol. 35, no. 5, pp. 408–420, 2011.

[3] L. Yin and Z. Wang, "Measuring visual enclosure for street walkability: Using machine learning algorithms and google street view imagery," *Applied Geography*, vol. 76, pp. 147–153, 2016.

[4] F. Moura, P. Cambra, and A. B. Gonçalves, "Measuring walkability for distinct pedestrian groups with a participatory assessment method: A case study in lisbon," *Landscape and Urban Planning*, vol. 157, pp. 282–296, 2017.

[5] L. Yin, "Street level urban design qualities for walkability: Combining 2d and 3d gis measures," *Computers, Environment and Urban Systems*, vol. 64, pp. 288–296, 2017.

[6] A. Frackelton, A. Grossman, E. Palinginis, F. Castrillon, V. Elango, and R. Guensler, "Measuring walkability: Development of an automated sidewalk quality assessment tool," *Suburban Sustainability*, vol. 1, no. 1, p. 4, 2013.

[7] Y. Rubner, C. Tomasi, and L. J. Guibas, "The earth mover's distance as a metric for image retrieval," *International journal of computer vision*, vol. 40, no. 2, pp. 99–121, 2000.

[8] E. Leslie, N. Coffee, L. Frank, N. Owen, A. Bauman, and G. Hugo, "Walkability of local communities: using geographic information systems to objectively assess relevant environmental attributes," *Health & place*, vol. 13, no. 1, pp. 111–122, 2007.