Radar in UAV obstacle detection and height navigation

Helder Alves Pereira
helder.a.pereira@tecnico.ulisboa.pt

Instituto Superior Técnico, Lisboa, Portugal

December 2023

Abstract

The world of electronics does not stop to grow. Many technologies benefit from these advances, such as radars. The tendency for these devices is to become smaller, cheaper, accessible and to provide known solution out of the box, such as point clouds. It's characteristics allow to know, with some noise, where objects are and their velocity relative to the radar, even in environments where conventional sensors have difficulties to operate. This work proposes an approach to use these devices in autonomous aerial vehicles (UAVs), aiming to discover the height relative to the ground and to know the objects ahead.
Keywords: Radars, Unmanned Aerial Vehicle, height estimation, object detection.

1. Introduction

Since the development of remotely controlled pilotless aircraft to attack Zeppelins in the First World War to today, Unmanned Aerial Vehicles (UAV) technology has evolved towards automation and others areas. Today, UAV's can fertilise crops, deliver packages, transport donor organs to risky and against time surgeries, work as cameras in the film industry, supervise and monitor areas, perform aerial light shows for amusement, perform battlefield missions, inspect buildings and power lines, arrive and search autonomously areas of difficult or dangerous access (such as Ingenuity, in Mars).

One of the many necessary fields to tackle is perception of the environment around them for the purposes of the assigned mission. For this task usually a depth sensor and a LiDAR or simple cameras are used.

However, sometimes these sensors may fail in certain conditions, such as LiDAR or depth cameras in presence of smoke, dust or fog [1]. Ideally many sensors should be used to complement each others flaws. In this case, radar is a sensor that can perceive objects even when in presence of fog or smoke. These sensors use electromagnetic waves that can travel through these elements. They have already been used in health context to monitor patients [5, 6], detect and track people [3]

2. Objectives

This work aims to propose a solution to use the pointcloud of a radar for space perception as a standalone technology. In other words, the main objective is the exploitation of the radar pointcloud as an input of space perceiving system for an UAV, in which can be divided into:

1. obstacle detection: necessary for perceiving the environment, given the pointcloud of a mmwave radar;

2. height information: necessary for the UAV to keep a safe distance from the ground or the distance requested for the UAV objectives.

3. Background

3.1. Radar

The radars that this work will use are often called mmwave systems since the working wavelength of these devices are in the order of millimetres. They fall in a category called MIMO radars, meaning Multiple Input Multiple Output in regards to the quantity of antennas that receive and emit power. These devices are small, and some can process the data automatically and concede a point cloud with the velocity of the points towards the radar.

These devices are often noisy and suffer from multiple path reflections, as shown in Figure 1.

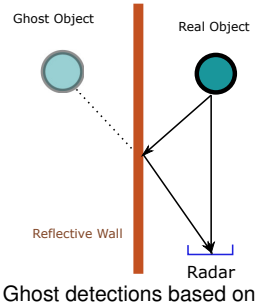

Figure 1: Ghost detections based on reflections

The radar readings have the following patterns:

- There can be multiple reflections of the transmitted signal, so that the received signal is not directly reflected from one object, as shown in Figure 1. However, reflections are not perfect and the reflector object can also be seen by the radar. In the end, the ghost object is always further than the reflector. This phenomena depends on the power of the emitted electromagnetic wave and it's more noticeable in enclosed spaces with many objects.

- Locations with objects contain more points and more consistently over time than empty areas.

3.2. Clustering - DBSCAN

For a set of data points disposed in space, sometimes is required to perform a data analysis in terms of its distribution along some criteria. There are many algorithms, such as Global Neirest Neighbours, K-means, amongst others.

One algorithm that will be often used in this work is the Density-based spatial clustering of applications with noise (DBSCAN). This algorithm groups data based on density according to a distance criteria defined by the user. For clustering based in the euclidean distance, it's only necessary ϵ and min_{pnt}. The parameter ϵ defines how close points have to be between each other to be considered neighbours. The min_{pnt} parameter defines the minimum number of points that a cluster needs to hold. The final effect is that it's possible to cluster data without initially specifying the number of clusters and points without a cluster are discarded.

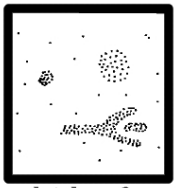

Figure 2: Visual examples of clustering from [4]

3.3. Occupancy Grid Map

To establish the regions of the environment that are free for navigation, to recognise regions, locations for some form of navigation or motion planning it is necessary to map the region where the robot is placed. For this case, composite-space maps allow a compact and a geographical representation of space.

Since the information given by the radar is a point cloud, for such scenario, a capable and known solution is to use occupancy grid maps. The assumptions of this method are:

- The world is discretized into rigid and static cells.

- Each cell in the grid represents a binary random variable of its probability of being free or occupied.

For the following algorithms 2 and 1 consider the following notation:

- N is user defined and $N \in \mathbb{N}$
- M is user defined and $M \in \mathbb{N}$
- m = the map composed by a 2D grid of $M \times N$
- $z_{1:t}$ = set of all measurements up to time t
- $x_{1:t}$ = the path of the robot defined through the sequence of poses

To map is to find an estimate of the probability $p(m|z_{1:t}, x_{1:t})$ where $m = \{m_1, m_2, \dots m_{M \times N}\}$. In practice the number of grid cells $N \times M$ might be very large and the calculation of the posterior probability might be intractable. Also the dependence between neighbouring cells might be ignored, resulting in the objective of calculating $p(m_i|z_{1:t}, x_{1:t})$ for every cell m_i of the map m.

The aim is to find the state of occupation for all the cells m_i in the map. To do this, each cell m_i is attached the probability of a binary occupancy value, 1 for occupied and 0 for free. To avoid instabilities for probabilities near zero, the representation used is the log odds form, as seen in equation 1.

$$l_{t,i} = log \frac{p(m_i|z_{1:t}, x_{1:t})}{1 - p(m_i|z_{1:t}, x_{1:t})} \tag{1}$$

The initial condition $l_{0,i}$ according to equation 1 is

$$l_{0,i} = log \frac{p(m_i = 1)}{p(m_i = 0)} \tag{2}$$

The inverse measurement model specifies a distribution over a binary state [8].

Having in mind the equations 1 and 2 it is possible to write equation 3

$$l_{t,i} = l_{t-1,i} + InverseSensorModel(m_i, x_t, z_t) - l_0 \tag{3}$$

From a simplistic point of view, the algorithm inverse range measurement model tries to find the cells m_i that are inside the sensor cone to mark them as free or as occupied, according to the readings of the sensor. The calculations used in the following algorithm 1 comply with the notation exposed in Figure 3 and in the following bullet list:

- α - thickness of obstacles;
- β - angular width of the sensor beam;
- (x, y, θ) - sensor pose;

- r - range for the centre of mass of cell m_i
- ϕ - bearing for the centre os mass of cell m_i
- θ_j - orientation of sensor j
- z^k - measurement k
- z_{max} - maximum possible measurement of the sensor
- assume $\theta = \theta_{j,sensor}$

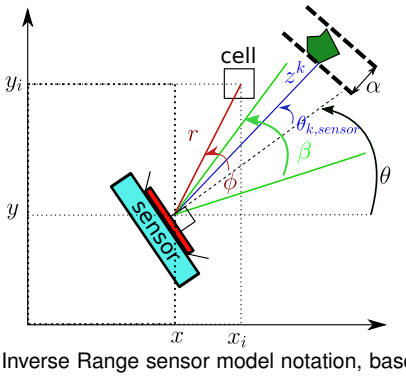

Figure 3: Inverse Range sensor model notation, based on [7]. Here a radar is looking at a green object. The notation in this figure is used in algorithm 1.

Algorithm 1 Inverse Range Sensor Model

Let x_i, y_i be the center of mass of m_i
$r \leftarrow \sqrt{(x_i - x)^2 + (y_i - y)^2}$
$\phi \leftarrow \text{atan2}(y_i - y, x_i - y) - \theta$
$k \leftarrow \text{argmin}_j |\phi - \theta_{j,sens}|$
if $r > \min(z_{\max}, z_t^k + \alpha/2)$ *or* $|\phi - \theta - \theta_{k,sens}| > \beta/2$ **then**
  return l_0
end if
if $z_t^k < z_{max}$ *and* $|r - z_t^k| < \alpha/2$ **then**
  return l_{occ}
end if
if $r < z_t^k$ **then**
  return l_{free}
end if

Having in mind the equation 3 and the algorithm 1 it is possible to write the algorithm 2.

The recover of the final probability is in the next equation 4

$$p(m_i|z_{1:t}) = 1 - \frac{1}{1 + e^{l_{t_i}}} \tag{4}$$

Algorithm 2 Occupancy grid map

for all cells m_i **do**
  if m_i in perceptual field of z_t **then**
    $l_{t,i} \leftarrow l_{t-1,i} +$ inverse_range_sensor_model(m_i, x_t, z_t)
  else
    $l_{t,i} \leftarrow l_{t-1,i}$
  end if
end for
return $l_{t,i}$

4. Object Detection

In a workplace $\Upsilon \subset \mathbb{R}^3$ an entity must travel through Υ. The entity is a UAV and the available inputs for the radar in the UAV to perceive the world are the following:

- 2D pointcloud from a mmwave radar - concedes spatial information. The K points in instant t are $z_t = \{z_t^1, ..., z_t^K\}$;
- speed of each point towards the radar. The measure z_t^i is a vector $[x\ y\ v_d^i]$;

The characteristics of the input information z_t are the following:

- The locations with objects contain a higher density of points than empty locations;
- Points represent locations where the radar is receiving power. This can be real reflections of ghost reflections as exposed in subsection 3.1. However, the discrimination of ghost objects won't be tackled in this work;
- The radar suffers from random noise, creating random points in random locations but with less density than locations with objects.

For the sake of simplicity, it is considered that the physical and scattering characteristics of the objects in the workplace Υ don't change over time. The goal is to create a system that can estimate the position of the objects shown by the noisy readings of the radar, having in mind that objects might be static or dynamic relative to Υ.

For the proposed solution, first lets consider a scenario where a group of N particles are disposed in a working environment Υ_S with an uniform distribution, for simplicity, at time $t = t_0$, with $p_{t_0} = p_{t_0}^1, ..., p_{t_0}^N$. These particles can move in 2D with a linear motion. The definition of each particle p_t^i will be $p_t^i = [x\ y\ v_x\ v_y]$. For a time step $\triangle t$, the motion model for each particle is in the equation 5.

$$p_{t+1}^i = \begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix} = f(p_t^i, \triangle t) = \begin{bmatrix} x + v_x \cdot \triangle t \\ y + v_y \cdot \triangle t \\ v_x \\ v_y \end{bmatrix} \tag{5}$$

If a particle contains at least one reading in its vicinity of a pre-determined radius called g_d, said

particle could announce that it might have felt noise or an object inside it's vicinity.

However, if such reading continues to occur in following radar readings, said particle has an increased confidence that an object is nearby. The opposite case is the lack of readings, increasing the belief of an empty surrounding. For better understanding, all the elements exposed so far are in Figure 4.

For now, the vicinity is defined as a simple circle of size g_d. To keep track of the frequency of detections in its vicinity, it is created a $score_t = \{score_{p_t^1}, ..., score_{p_t^N}\}$. In short, each $score_{p_t^1}$ will work as the following: If over time the particle continues to see readings in its vicinity, two situations can be the culprit:

- the particle is accompanying an object with success;
- the vicinity of the particle is too big and it is detecting multiple objects due to their close proximity;

. Either way, the particle has an increased belief that it might be near an object.

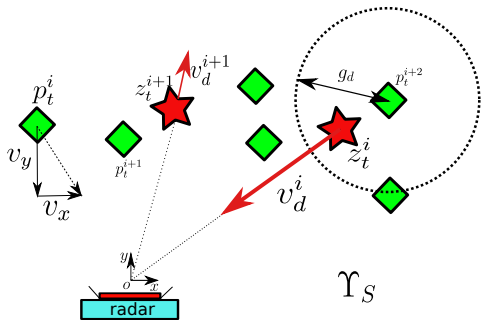

Figure 4: Simple scenario in Υ_S with a point cloud of the radar composed by two points (red stars). Some particles (green diamonds) are displaced in Υ_S. A visual example of v_x, v_y, v_d and $gate$ are exposed.

However, a problem still remains so far: it some point z_t^i is noise, all the neighbour particles will assume that they saw a point corresponding to an object. However, since it's noise, that point won't remain in the next iteration. To countermeasure this, an T_β is created. If surpassed, it is considered that the particle is successfully accompanying an object.

The higher the number N, the higher the number of particles and the required calculations, increasing the computation effort. Since it's considered that Υ_S is dynamic, any object can appear in scene, so particles must be available to accompain new objects. To resolve this without increasing N over time, particles with a low $score_{p_i}$ can be ressampled to new values of position and velocity, since they are not accompanying any object. For such task, a T_α is created to compare when should a particle be sampled again.

Given the previous explanation, the algorithm 3 can be written:

Algorithm 3 Voting Dynamic Particle Filter

```
Generate N particles with velocity, position
Generate N particles with score at zero
while Working do
    Read radar and sensor data
    Predict where particles will be according to the model in equation 5
    for i = 0; i < N; i ← i + 1 do
        Find closest measurement d_k to particle p^i
        if d_k < gate then
            Increase score_{p^i}
            if score_{p^i} > T_β then
                Consider particle as successful tracker
            end if
        else
            Decrease score_{p_i}
            if score_{p^i} < T_α then
                Ressample particle p^i and reset score_{p^i}
            end if
        end if
    end for
end while
```

If an area contains a cluster composed by a high density of particles considered as successful trackers, then there is a high probability that they are accompanying an object.

So far, the algorithm 3 has a close resemblance to the occupancy grid map exposed in subsection 3.3. Now the group of cells m represents a group of possible cells that are distributed in four dimensions, with a quadruplet of values $[x\ y\ v_x\ v_y]$ as stated in equation 5. If L_x,L_y, L_{v_x} and L_{v_y} are considered the number of admissible values for each dimension of p^i then the number of cells m_i, now stated as p_t^i, is $L_x \times L_y \times L_{v_x} \times L_{v_y}$ where $m_i = 1$ means a successful accompanying of some object and $m_i = 0$ means a non successful accompanying of an object. For a fine grained and large space Υ_S the value of $L_x \times L_y \times L_{v_x} \times L_{v_y}$ is impossible to implement in terms of memory. Having this in mind, for an frame $t = t_i$, only a small subset of the possible states m_i of m is considered. This subset corresponds to the group of particles p.

Since any object can appear in any given time with any velocity, to increase even more the possible states, particles p_t^i with a low $score_{p_t^i}$ are resorted, to allow the estimation of a new possible state.

Now, the probabilities are not recovered but the values $score_{p_t^i}$ are simply compared to T_β to state if the particle p_t^i is a successful accompaniment of an object.

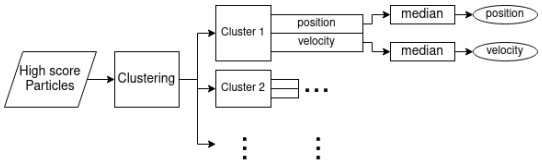

Figure 5: Schematic of the final pipeline of the proposed solution

Since regions with objects will have more particles and the number of clusters is unknown, the DBSCAN is an capable solution. To each cluster, the following calculations can be applied:

- A median to the locations of the particles of each cluster to concede a better tolerance to outliers;
- A median of the speed of the particles having in mind the $score_{p_t^i}$, since higher $score_{p_t^i}$ means that the particle has been accompanying an object for more iterations, meaning that it has a better estimation of the velocity of the object.

Now there is a list of variables to tune:

- number of particles N;
- Limits of Υ_S;
- gate distance g_d;
- threshold T_β;
- threshold T_α;
- limits of the uniform distribution that generates the values of x,y,v_x and v_y of each particle;
- ϵ of the clustering stage;
- min_{pnt} of the clustering stage;

The physical limits of Υ_S are defined by the acceptable performance of the radar and it's mission. For example, if the radar can't give readings beyond 2 metres, then its not required for Υ_S to have a depth of 3 metres.

The values of N, g_d, T_β, T_α depend heavily from the size of Υ_S and the performance of the radar, for example:

- the values of N and g_d must allow for the full coverage of Υ_S since it is not known when, where and how many objects will appear in Υ_S at time t;
- If the random noise has a high density, close to the value of real objects then the value of g_d must be decreased and T_β must be increased.

In short, for each requirements of a mission, for each radar, environment Υ with its unique characteristics, different parameters must be used.

5. Height Estimation

In a workplace $\Upsilon \subset \mathbb{R}^3$ an entity must travel through Υ. The entity is a UAV and the main purpose is to estimate its height for purposes of safety or objectives of its mission. For that, the problem is a simple and reliable exploitation and usage of a small mmwave system.

The list of available input information is:

- 3D or 2D pointcloud from a mmwave system looking down - concedes spatial information;
- speed of each point towards the radar;
- linear and angular measurements from an IMU unit - concedes information about the attitude of the radar;

The output of the system is a value with the information about how high is the UAV.

Radars are known to be used in aeroplanes to estimate the height where they are. However, small mmwave radars working with many objects in close proximity can create multiple path reflections, as depicted in Figure 1. In low altitudes this can constitute a problem, as exemplified in Figure 6. The readings representing the real ground are the red stars, the fake readings from the multipath reflections between the ground and the UAV are the yellow stars.

In Figure 6.b there is two represented paths:

- At red there is the real path. The electromagnetic wave leaves the radar, bounces in the ground, bounces in the UAV, bounces again in the ground and reaches the UAV.
- At yellow there is the perceived path by the radar.

The case shown in Figure 6 invalidates simple approaches such as an average or median of the readings given by the radar. It is possible to prevent cases as shown in Figure 6 by using dedicated materials [2]. Other counter argument is that the power of the mmwave radar is to weak or the UAV does not possess enough reflective area to create such scenario. However, that might not always be the case and a simple solution can be created to prevent it.

Starting from a scenario where the radar is parallel to the gravity like in Figure 6, the solution is composed by the following steps:

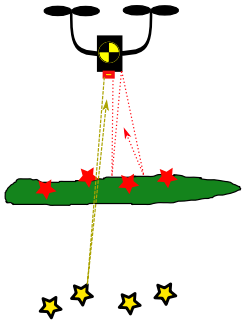

Figure 6: Illustrative example of problems with height estimation. At low heights, multipath reflections create the illusions of multiple objects. Real readings are at red. Fake readings are at yellow

1. Consider only the readings inside a cylinder C of radius R with its axis A from the radar to the ground. The axis A should be parallel to the gravity by using the IMU;
2. Cluster the remaining readings by distance to the radar using the DBSCAN algorithm;
3. Choose the closest cluster to the radar and consider it the ground.
4. Calculate the y median of the closest cluster

A cylinder is considered since its a simple method to cover a large area at low altitudes without knowing the altitude. The minimum radius R should be the radius of the UAV. An extra margin can be added, depending of the intentions of the mission of the UAV.

The clustering step is made with the DBSCAN since this algorithm does not need to know the number of clusters. The mean of the distances of the closest cluster is the final value.

Now there are only two variables to estimate, and these are from the DBSCAN algorithm: min_{pnt} and ϵ. For different radars or working distances the tuning of these values might be different.

6. Results

In this section the data comes from the radar AWR1443BOOST and the ground truth comes from the Intel Realsense D435i depth camera, which will be ofter refered as RGBD.

6.1. Moving against car

For this experiment, a radar is going to move against a car. The movement is constant in the x axis and not constant in the y axis. The purpose of this experiment is to prove that the algorithm 3 can detect and know where an object is with the UAV in movement. This scenario simulates a possible crash.

For this experiment, the radar was tuned to concede readings with the best range resolution up to 10 metres.

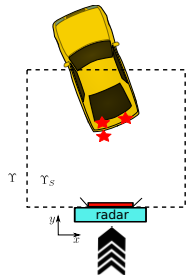

Figure 7: Bird view of the performed experiment. The radar is moving towards the car, in the y axis. The origin of the axis for the measurements is in the radar.

The parameters to tune from the algorithm 3 developed in Section 4 are:

- Given the the admissible values where the particles p will reside relative to the radar are limited to:
  - minimum of x: $-5m$
  - maximum of x: $5m$
  - minimum of y: $0m$
  - maximum of y: $6m$
  - minimum of v_x: $-4m/s$
  - maximum of v_x: $4m/s$
  - minimum of v_y: $-4m/s$
  - maximum of v_y: $4m/s$
- the number of particles N must be able to cover all the area and concede multiple velocities to be able to accompany possible objects in real time: $N = 10000$
- T_α = -1
- T_β = 3
- g_d = 0.2 metres.

The data for ground truth from is displayed in Figure 8 as blue dots, representing the car. Later correction and calibration to collocate the data in the same referential was made on software in Matlab. A time frame of the collected data is disposed in Figure 8.

The data shown in 8 is exposed to simplify and compare where the radar readings z_t^i are and where the successful particles p_t^i are. These shown particles all have a score higher than T_β. The particles with a lower score than T_β are hidden since they will not contribute to any accompany of any object. For the same time snapshot, these are shown in Figure 9 just for demonstration.

In Figure 9 its possible to see that even inside the cluster of successful particles there are particles with a score lower than T_β. That happened because these particles didn't have enough frames to

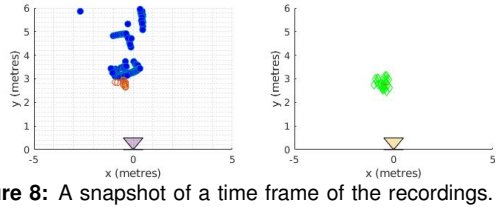

Figure 8: A snapshot of a time frame of the recordings. Both graphs represent the same time and scenario. At left there is the radar readings represented by brown circumferences each one with their velocity v_d^i represented radially from the centre of the reading to the radar. The blue balls are the car, representing the ground truth. The radar is represented by the triangle at the origin. In the right there is the same scenario but showing the successful particles with a score higher than T_β as green rhombus.

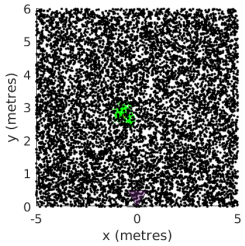

Figure 9: This graph is the same as the right graph in Figure 8 but now the particles with a score lower than T_β are shown as black dots.

accumulate enough score. That occurs when particles just arrived close to the cluster or because they where resampled into there.

For computational purposes of studying the errors and the performance of this algorithm and its configuration, the visible cluster will be tracked by proximity. In other words, for a frame in time $t = t_j$, each cluster will be identified by proximity of the list of clusters in time $t = t_{j-1}$. Although in this case it is only being measured the movement against a car, this approach allows to track clusters of particles in more complex scenarios with more objects. The clustering considers that $\epsilon = 0.2$ and $min_{pnt} = 2$.

The final values of position and velocity are shown in the next Figures 10

Some statistics show are shown in Table 1. The

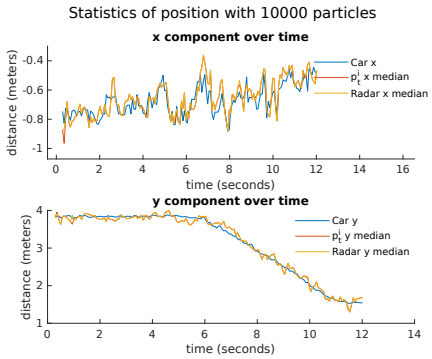

Figure 10: Position statistics of the tracked cluster of particles of the car.

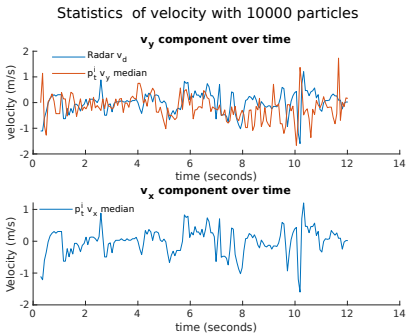

Figure 11: Velocity statistics of the tracked cluster of particles of the car.

shown errors are the RMSE since this measurement is sensible to outliers.

Table 1: Positional RMSE in metres from the performed experiment.

		Car		Radar	
		x	y	x	y
Radar	x	0.0667	–	–	–
	y	–	0.077	–	–
Particles	x	0.0662	–	0.0073	–
	y	–	0.0775	–	0.0079

From Table 1 its possible to see that the RMSE between the position of the particles and the radar is one order of magnitude smaller than the errors between both radar and the measuring of the car. This means that the particles are very reactive to the readings of the radar and follow then precisely.

Since the movement is not fully linear, when analysing the velocity data from Figure 11 it is not expected to have a constant value for velocity in the y axis. Overall, the values of the velocity v_x are very noisy, achieving errors that can reach $1m/s$ while the horizontal movement in the recording of the experiment is closer to zero. At the same time, if the reference of the velocity v_y is established to be v_d then RMSE can be calculated to be RMSE(v_x, v_d)=$0.4435m/s$. This error is very high, specially for a movement of y from 4 metres away to 1.5 metres in 7 seconds.

One observation that can be done between the velocities v_x and v_y of the particles is that they are very similar. In fact, the average of the differences between v_x and v_y is $0.1531m/s$. After realising this, the parameters of the tested algorithm where modified to create less reactive particles, aiming to enforce scarcity of particles to the clusters with a score bigger than T_β. To achieve this, the following values where changed:

- $g_d = 0.1m$
- minimum of v_y and v_x: $-2m/s$
- maximum of v_y and v_x: $2m/s$

The result is a smaller average error between v_x and v_y of $0.1355m/s$. This indicates that the implementation might have problems in the number generation numbers. Unfortunately, this error was detected too late for correction of the software implementation. Further analysis of the implementation is necessary to extract better and more trustworthy conclusions.

6.2. Tacking off movement

Since the sensors are not yet implemented in a UAV to test measurements against a ground, an equivalent scenario had to be created. Holding up the sensors looking down was not a feasible approach since there where interference from the person holding the sensor recording apparatus. It was decided that measuring the distance perpendicular to the wall was a possible simulation of measuring a plane ground.

The parameters to tune are:

- ϵ of the DBSCAN stage;
- min_{pnt} of the DBSCAN stage.

The only parameter that depends on the UAV or context of the mission is R. In the following experiments, R = 2 metres.

The radar was tuned to be able to concede readings with the best resolution possible to at least 6 metres.

To simulate a variation of altitude of an UAV, a taking off was imitated for the parameters where the radar was tuned. For such, the sensor box is placed facing and close to a wall. After a few seconds, the person holding the sensors moves away from the wall with a constant velocity, as depicted in Figure12.

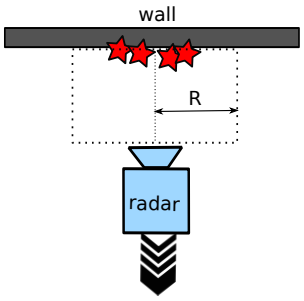

Figure 12: Top view of the scenario. The red stars are the readings from the radar. The wall is the grey rectangle. The radius of the cylinder C is depicted as R. The black arrows display the performed movement

A time frame of the data collection is shown in Figure13.

In Figure13 each segment of the grid corresponds to 1 metre. In this frame it's possible to see that at about 2 metres of distance, the radar has a bigger field of view than the depth camera.

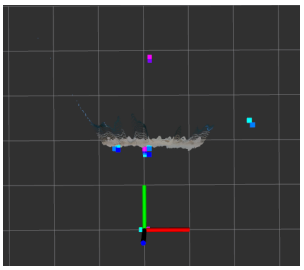

Figure 13: Top view of the collected data of the scenario in the software RViz. The colourful large points are the radar readings. The fine grained points are the readings of the depth camera. The radar is collinear to the green axis.

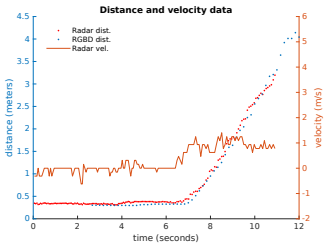

Figure 14: Display of the collected data. The radar distance, velocity and the ground thruth given by the depth camera are exposed.

The example frame depicted in Figure13 was chosen because it's possible to view the phenomena o multiple path reflections. At about 4 metres of distance from the radar (double of the distance to the wall), it's possible to see a group of points that correspond to a multiple reflection. The electromagnectic wave that generated this ghost object took the following path:

1. Radar to wall;
2. Wall to sensor box where the radar is mounted (and receiving antennas of the radar);
3. Sensor box to wall;
4. Wall to radar receiving antennas.

After collecting the data, the values of ϵ and min_{pnt} where tuned for the best performance between 0 and 6 metres. Analysing the point cloud given by the radar, the final values where:

- ϵ = 0.2
- min_{pnt} = 4

The collected and treated data is shown in the graph of Figure14. The median of the velocity given by the radar points used for height estimation is also displayed in the graph, in the right axis.

In the Figure14 it's possible to make the following observations:

- The start and stop of the recording of the depth camera was later relative to the radar;

- At the beginning of the experiment, the radar was resting at 0.32 metres from the wall;
- The time where the radar started to move away from the wall was at 7 seconds;
- Since the radar was tuned for the best distance resolution, the resolution of the velocity was deprecated.
- The moment the radar starts to move, the velocity indicated by the radar increases, indicating that objects moving away have a positive velocity.
- In this recording, the sampling frequency of the radar is faster than of the depth camera;

To test the quality of the data while moving, the data from 5 seconds until the end was selected. For this interval the following values where calculated:

- The root mean squared error $RMSE$ of distance between radar and the RGBD = 0.1132 metres
- Linear fit radar r^2 = 0.9886
- Linear fit RGBD r^2 = 0.9879
- Radar fit: $y = 0.7034x - 4.5066$
- Depth camera fit: $y = 0.7378x - 4.8805$
- median of velocity = $0.9358m/s$

The values of r^2 defines how well the data is fitted to the desired model. In this case, the closer to 1, the better. Since the values of r^2 for the linear regression of the radar and movement and depth camera are close to 1, then there is a high degree of confidence that the movement was indeed at a constant velocity. In that case, the speed was extracted from the slope of the linear regressions, considering the speed indicated by the depth camera as the ground truth. The relative errors are in table 2:

Table 2: Relative velocity errors considering the depth camera as ground truth.

	Value(m/s)	Ground Truth
Radar points	0.7034	4.67%
Radar speed median	0.9358	26.84%
Depth camera	0.7378	–

The absolute error between the radar velocity is the biggest, as expected, since the resolution of the speed was deprecated to give a better distance resolution.

For a constant movement between 0.32m to 5m, the final conclusions are:

- The $RMSE$ is 0.1132m, which is one order of magnitude of difference relative to the measured distance.
- According to the table 2, the relative speed to the ground (wall in this case) is better estimated from the position of the points of the point cloud than the instantaneous velocity given by the radar.
- Depending on the mission, for this radar, if the user would need a better quality for the instantaneous velocity of the radar, he would have to deprecate the distance resolution.

.

The statistics for the time interval $t = [5, 11]$ seconds are in table 3.

Table 3: Final values if all points inside inside the cylinder C are considered for the time interval $t = [5, 11]$.

	Radar point cloud	RGBD point cloud	Radar speed
RMSE(m)	0.2013	–	–
r^2	0.9738	0.9879	–
Velocity(m)	0.6931	0.7378	0.9358
Relative error with eps=20m	6.06%	–	26.84%
Previous relative error (eps = 0.2m)	4.67%	–	26.84%

In table 3 it's possible to see that the $RMSE$ increased and the relative errors for the distance indicated by the radar point cloud increased as well. Since some noisy points are now considered for calculations, it's natural that the errors increased. However, the value of the speed stayed the same. At least one of tree situations can be the culprit:

- The median is not letting the outliers have a greater effect in the final calculation;
- The outliers are "only outliers" relative to the position and not to the value of the speed;
- This experiment is not noisy enough to expose the true performance of the proposed solution.

7. Conclusions

The industry of radars will keep growing in the following years, and the small radars will keep improving, allowing miniaturisation and implementation in UAVs.

In this work two simple and intuitive methods where designed to use the point cloud from a small mmwave radar in a UAV. The estimation of objects

was deduced intuitively to be of general purpose for any small mmwave radar that creates point clouds. If precise measurements of location are not available, the detections of static or dynamic objects and their follow up is independent of the movement of the UAV since the developed algorithm adapts to movement. Further analysis is necessary to improve the implementation of the algorithm in software. The estimation of height was performed with also an intuitive approach to concede a simple understanding for general implementation. The height estimation was tested simulated against a wall, proving that for block surfaces the distance measurement works for the considered range.

One scenario that was not tested and the provided solution can't solve is the case where if a ground is composed by a slope made of a perfect reflector, depending on the angle of the slope and area considered for height calculation, it can exist the situation where no power will be sent back to the radar, not producing radar readings. A new solution must be researched for a standalone usage of the radar.

8. Future Work

A lot of work can still be done:

- First and foremost, added testing to the implementation needs to be done in more challenging scenarios and with different radars;
- For each cluster of particles p_t^i in the space Υ_S, the component of the velocity of the particles towards the radar can be improved by the velocity v_d^i given by the readings of the radar. The velocity v_d^i can be "mapped" with the same logic onto the particles p_t^i, possibly conceding better results. However, such approach must account the limitations of the radar (mostly noise) in use and find a good way to compare the velocities v_d^i of the point cloud and the velocities of the particles to quantise the similarity between a point and a particle.
- The generation of particles followed a simple uniform distribution. This was a simplification for the implemented algorithm. However, the creation of particles might follow another distribution if there is context or some information from readings of other sensors, possible leading to a perform a better generation and performance.
- This work tries to give information of where objects are relative to an UAV. However, given the limitations of the radar, a multi-sensor solution is advisable. In such case, a system should be constructed to decide which information from each sensor is trustworthy.
- The proposed solution for detections of objects considers only the value of the point cloud. A different working pipeline could improve if given the information of the power that originated the points that are given by the radar;
- This work considers a usage of a sensor for height altitude and one sensor for object detection. For such scenario, two of these sensors have to use different frequencies. An interesting approach could be the usage of a 3D radar to see the ground and the environment ahead of the UAV at the same time. Depending of the radar and the quality of its 3D capabilities, a perfect angle would have to be chosen.

References

[1] M. Bijelic, T. Gruber, and W. Ritter. A benchmark for lidar sensors in fog: Is detection breaking down? *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 760–767, 2018.

[2] A. Choudhary, S. Pal, and G. Sarkhel. Broadband millimeter-wave absorbers: a review. *International Journal of Microwave and Wireless Technologies*, 15(2):347–363, 2023.

[3] H. Cui and N. Dahnoun. High precision human detection and tracking using millimeter-wave radars. *IEEE Aerospace and Electronic Systems Magazine*, 36(1):22–32, Jan. 2021. doi:10.1109/MAES.2020.3021322.

[4] M. Ester, H. P. Kriegel, J. Sander, and X. Xiaowei. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, 1996.

[5] B. R. Mostov K, Liptsen E. Medical applications of shortwave fm radar: remote monitoring of cardiac and respiratory motion. *Medical Physics*, 37(3):1332–1338, Mar. 2010. doi:10.1118/1.3267038.

[6] S. Pisa, E. Pittella, and E. Piuzzi. A survey of radar systems for medical applications. *IEEE Aerospace and Electronic Systems Magazine*, 31(11):64–81, Nov. 2016. doi:10.1109/MAES.2016.140167.

[7] M. I. Ribeiro, P. Lima, and R. Ventura. Mapping occupancy grid mapping. Instituto Superior Técnico/Instituto de Sistemas e Robótica - Autonomous Systems, 2021.

[8] D. F. Sebastian Thrun, Wolfram Burgard. *Probabilistic Robotics*. MIT Press, 2005.