1 Introduction

One of the most interesting features of an autonomous robot is its ability to navigate independently of human intervention through its environment to reach its destination without colliding into obstacles. The environments in which the robot may be required to navigate are typically categorized according to the level of prior knowledge on navigation or task space. The environments can be split into one of the following categories [1]: (1) known, (2) partially known, and (3) unknown environments. A known environment is specially designed for a robot to perform routine tasks in a factory or an office environment with accurate knowledge of it. On the other hand, an unknown environment represents the case when none of models or maps exists, or it can not even be accurately generated. Robots in this case operate purely in response to real-time sensor data or navigate under careful sensing and planning. This situation is often encountered when the robot explores the natural real world or artificially created world without prior knowledge. Finally, a partially known environment is somewhere between the previous two extreme cases. The environment information modeled to a certain extent is given to the robots, but it is insufficient to fully support task completion. Most applications for indoor or outdoor robot navigation, except fully unknown space exploration, are closely related to this situation. The simplest one is the case where just the floor or ground information on which robots navigate is given [2, 3]. In these applications, the robot detects obstacles on its navigation surface from 3D range data using a priori known ground information and uses the obstacle information for avoiding or recognizing them. However, for operations with a higher level than that of obstacle avoidance, the obstacle region detection on the 2D navigation map is not sufficient and we need 3D model of the environment.

In the literature there exist two fundamental paradigms for modeling the environments in front of the robot: the grid-based (metric) approaches and topological approaches. Grid-based approaches, such as those proposed by [5,6,7] and [4] and many others, represent environments by evenly-spaced grids. Each grid cell may, for example, indicate the presence of an obstacle in the corresponding region of the environment. Topological approaches, such as those proposed by [8,9,10] and others, represent robot environments by graphs. Nodes in such graphs correspond to distinct situations, places, or landmarks (such as doorways). They are connected by arcs if there exists a direct path between them.

In this paper we use synthetic aperture radar (SAR) to get data related to the environment, as well as that related to the different objects including on that environment since SAR is distance and weather invariant. Once we have the data related to the environment surface and that related to the different objects existing on this surface, we use splines to get the triangular mesh model of that surface and this is detailed in section 2; the control scheme summarizing the process followed in our work is shown in figure 1. The representation of the environment and the existing objects in form of triangular mesh allows the robot to locate the different objects and the target, and then search for a free path linking the initial position to the target which is tracked by the SAR sensor and this is discussed in section 3. Before all of these we deal with path planning of 2D and 3D environment using linear parametric curves (section 1).

2 Path planning with obstacles avoidance

We are concerning with path planning problem of an autonomous mobile robot operating in 2D and 3D environments with obstacles. The objective of this section is to find a collision free path from a given start position to a predefined target point. We explain the approach of path planning and obstacles avoidance using parametric curves initially by considering the robot as a 2D material point. It knows the coordinates of its current position and
the target. It is equipped with synthetic aperture radar sensor, which provides readings 360° around it. The robot can measure the distance to the closest obstacles which are within the sensor range and avoid them to reach the target following a smoothed optimal path.

### 2.1 The use of linear parametric curves

The use of parametric curves for path planning is very developed and largely used in computer graphics [12], in design and manufacture computer-assisted [11], [14], and [17]. Parametric curve has an inherent directional property which enormously reduces calculations.

This property have been successfully demonstrated in 3D path planning algorithm for an unmanned aerial vehicle (UAV) operating in cluttered natural environments [16]. In that work greater attention is placed on the computational complexity in comparison with other path-planning considerations.

In our work, a linear parametric curve is used for path planning, and then this curve will be smoothed around the control point. The linear trajectory connecting the initial to the goal points is examined in the first step. If the trajectory collide an obstacle, a control point is introduced between the initial and the target point and an intermediate connection point is created once again (see figure 2).

Control point space has a property in Free Space (FS) which defines a trajectory without collision. Free space is a surface belonging to the CPS (Control Point Space), but unoccupied by the image of the obstacles. The geometrical layout is a computing process. The complete construction of CPS is not necessary for the practical problems. The construction of the CPS depends on the manner in which the control points are defined. We used a circular workspace for control point definition, i.e. in polar coordinates (the radius $\rho$, and the rotation $\theta$); this method eliminates the interference checking calculations after smoothing (the flowchart explaining the method is given as an appendix).

### 2.2 Connection smoothing using NURBs

![Diagram](image1.png)

Figure 2: Path in form of a series of linear segments connected by curves.
The path obtained until now is discontinuous at the control point. To remedy this problem, we use the NURBs (Non Uniform Rational B-spline).

Many researchers have addressed the smoothness objective such as: [18] where an algorithm which generates a path with a low curvature was proposed, the authors used the smooth control laws proposed in [13] to follow the generated paths. In addition to [15] which addressed the path planning problem for a wheeled robot.

In our algorithm, initially we add only one control point; then we test for collision between QT and the obstacle. In case of collision, we have to insert another control point, and so forth.

In figure 3, the black spline shows the smoothed path to avoid the first obstacle (obs.1), and in order to avoid the second obstacle (obs.2) a control point is added to this spline. The refitted spline is shown on red.

It is clearly shown that the addition of a control point does not affect the global spline. The simulation is done using Visual Basic macro under AutoCAD software. Figure 4 is an illustration example where we can see clearly that obstacles may be of any shape, but circular objects must be circumscribed by polygon with maximum corners; otherwise the algorithm will be slow. In our simulation we used the non-uniform rational B-splines (NURBs) as they preserve the $C^2$ continuity of the trajectory which is very important for robot dynamic.

### 2.3 3D path planning

Now, we consider the planar path planning of virtual robot; thus we have to consider the width and the height of the robot which has to avoid 3D obstacles. In 3D case the problem can be reduced to a problem of two dimensions by projecting the objects on the plane containing the initial point, the target point and the control points.

The implemented algorithm creates three selection sets: the first set contains the virtual path, the second contains all the obstacles existing in the environment and the third set is initially empty; then the algorithm checks the interference between solids in the first selection set against those in the second selection set.

Once the algorithm starts checking for interferences, temporary interference objects are created and included within the third created selection set. In order to avoid the obstacle, the virtual path must be rotated around the starting point, $S$, by a small angle $\Delta \theta$ given by equation 1.

$$\Delta \theta = \sin^{-1} \frac{w}{ST}$$

(1)

Where: $ST$ represents the distance between the starting point and the target, and $w$ is the expected robot width.

After this rotation, the temporary interference objects that have been created during interference checking will be deleted. Thus the interference selection set becomes empty.

The algorithm rotates again by $\Delta \theta$ and checks for collision by testing the interference selection set, the linear path is in collision with obstacles unless the selection set is not empty. It keeps rotating until the interference selection becomes empty. When the selection set is not empty, the created interference objects must be removed before the next iteration. The iteration is limited for obstacles interfering with the linear curve.

When the linear path collides an obstacle, a control point must be inserted between the starting point and the obstacle.

To obtain the control point $Q$, all the obstacles must be projected into 2D workspace, thus we use the same procedure as explained in subsection 1.1; for the projection of the 2D to 3D data we use Principal Component Analysis (PCA) and Multidimensional Scaling (MDS). The interference checking must be carried out for the obstacles above and below the line $ST$, hence rotation is done either to the left or right. The path obtained until now is discontinuous at the control point. To remedy this problem, we use the NURBs (Non Uniform Rational B-spline) as explained in subsection 1.2. Figure 5 is an illustration example.

Figure 3: The addition of control point to the smoothed path.
3 Environment modeling

3.1 Environment sensing using radar systems

Our aim is to apply this approach of path planning to allow the robot to navigate in Sahara domain. Most of the robot missions in Sahara domain are carried out under extreme weather and poor visibility conditions. The probability of collision is very high, if the robot is not equipped with all-weather day and night obstacle detection capability. The increasing tendency of the robot accidents demands the development of all weather obstacle detection capability. Radar is the primary electromagnetic sensor, which posses sensing capabilities for detection and screening. Radar sensor systems create their own electromagnetic energy, which is transmitted towards the objects through the atmosphere, and is finally recorded by the sensor receiver.

One of the main advantages of radar sensing systems is that, they possess the capability to collect data during any time of the day or night. Radar systems transmit short bursts or 'pulses' of electromagnetic energy in the direction of interest and record the origin and strength of the backscatter received from objects within the system's field of view.

The obstacle detection system comprises the following subsystems.

- A radar sub-system, which emits a beam of laser energy, receives the returns from the objects and finally processes the returns and produces the relative data related to objects.
- A scanning sub-system scans the beam and provides information related to direction and range of the objects.
- A processor after receiving information processes the data relative to range, direction and size of the objects and displays it on screen.
- A display system, where the real time information will be presented.

Information extraction

In this step we take the real time information which is presented on the display system and we extract the model of the environment. First we use the linear parametric curves, as explained in section 1, to extract all the control points consisting the surface of the environment. As the surface of the environment is represented as a 3D object, we search for the control point following the x-y plan by moving with a small step, then we increment the step in the z direction and we search the control points in the x-y plan once again and so on. Figure 6 is an illustration example.

By applying this algorithm we obtain the triangular mesh representing our environment, this technique allows the partition of complex models into simpler components. Moreover, it results in pieces that are easy to process. The simulation results are obtained by using Visual basic tools under AutoCAD software.

The technique returns also the triangular mesh of each side of complex objects, and this can help as in obstacle avoidance as it allows the object’s shape recognition. Figure 7 is an illustration example; the blue points show the different sides.
The control points represented by the intersection of the triangles of the mesh, will be used to locate the objects existing on the environment and to indicate the free path. The free path is obtained by linking the control points, which do not contain any object, using NURBS. The obstacles existing in complex environment are avoided using the same algorithm as that explained in section 2.3.

4 Target tracking and identification

a. Target tracking

We can use spotlight mode radar to capture an image of a target. However, with uncertainty from measurement clutter, one needs to know where to point the radar, which is a slewing problem. In order to enhance track quality, the fusion of tracking and automatic target recognition information can aid in determining the number, type, and orientation of tracking objects to effectively point the radar.

Measurement uncertainty might result from sensor errors, partial scans, and incorrect pointing of the radar. Target uncertainty results from incomplete knowledge and affects the prediction of the number of existing objects. Incomplete knowledge can be represented as a set of information for what is known, appended with a component that captures the unknown information. Sometimes the known information is improperly assessed and the most likely probability is assigned to the target, as in the case of misclassification of targets. By capturing unknown information from other sensors we can capture incomplete knowledge. This problem is remedied by the use of sensor management [7], this help to obtain all the information about a target for classification and ID at each time update.

b. Target identification

SAR target ID is similar to image processing techniques [1]. The difficulty with getting a SAR image is that targets need to be moving at a slow speed, or one in which the sampling time can collect the target within the time allocated. Typically, a SAR image takes up to 10 seconds to construct. For sampling purposes, the target should be moving at a speed such the movement is not greater than 5 km per hour [10]. If the target is moving faster than this rate, a portion of target can be assessed from the Doppler shift and is usually considered as high-range resolution radar. Obtaining a SAR image is much like a camera; however, the quality of returned image is not as sharp as a camera and is usually blurred from inherent Doppler processing of the image.

The identification of the target from the other objects can be done by creating a data base containing the information of all the existing objects, then for each object we compare its information and its characteristics with that of the target until we identify which object is the target, and the remaining objects will be considered as obstacles.

Figure 6: The search for control points.

Figure 7: The decomposition of the environment into its triangular mesh.
5 Conclusion and perspectives

The paper has proposed a method to solve the path planning problem of an autonomous mobile robot operating in 3D environment. The algorithm generates a free collision path starting from the initial position to the target position using the parametric curves characteristics. The radar sensor returns the data related to the environment and the obstacles, and then we used the triangular mesh for modeling the environment and localize the obstacles. And this allows the robot to navigate in any environment (simple or complex one) with obstacles avoidance. The target is tracked and identified using the information returned by radar sensor. Furthermore the generated path is smoothed using the NURBs. As an assumption the target must be stationary object. For future work we try to cover any type of target stationary or moving ones, and we try also to design a robot-user interface using Visual Basic software.

References