3D Simultaneous Positioning and Mapping in Dark, Closed Spaces with an Autonomous Flying Robot

Abdülkadir Çakır[1], Seyit Akpancar[2]

[1]Department of Electrical and Electronics Engineering, Faculty of Technology, Isparta University of Applied Sciences, Isparta, Turkey
E-mail: abdulkadircakir@isparta.edu.tr

[2]Department of Computer Programming, Atabey Vocational School, Isparta University of Applied Sciences, Isparta, Turkey
E-mail: seyitakpancar@isparta.edu.tr

Abstract: This study describes the steps needed to produce the required hardware and software for the 3D mapping of dark, rugged, and closed spaces such as caves, underground cities and mining pits through a DJI Matrice 100 Flying Robot Platform that has a high bearing capacity of 3600 grams and has no limit of movement in bumpy, hollowed or sloping spaces. In order to obtain the obstacle information around, during the autonomous movement of the air robot used within the scope of this study, 5 ultrasonic sensors – right, left, front, top, bottom – were used. A servomotor driven electromechanical equipment that will be used on the z-axis movement of Hokuyo UST-20LX laser sensor, which provides data in 2D, was developed to help the air robot map its environment in 3D during the autonomous movement. The control of the hardware developed and used in this study is carried out by Robot Operating System (ROS) nodes written in C++ programming language. The mapping studies were carried out by operating the robot autonomously in caves within Atabey District of Isparta Province, Turkey, at the coordinates of 37°53'41.8"N 30°32'58.5"E and 37°53'39.0"N 30°32'42.3"E. It is shown that the 3D maps produced by the system, are realistic and substantial.

Keywords: flying robot; 3D; simultaneous; mapping; dark spaces; closed spaces

1 Introduction

The mapping of space, such as caves or underground cities of an unknown shape, is mostly done using traditional methods, such as, takings manual measurements. However, when it is dangerous for people to enter a closed area, either mapping is not performed or performed by taking risks and entering the closed area.

In the mining accident, which happened in Turkey, on May 13, 2014 and took part in literature as the "Soma Mine Disaster", rescue activities were hampered by gas from the fire in the mine and by the lack of a known plan for the mine [1]. The importance of using autonomous robots actively emerges in such events.

Within the scope of this study, a fast and low-cost system which can operate autonomously in dark, rugged, and closed spaces such as caves, underground cities, and mining pits without any movement limits (bumpy, hollowed, sloping spaces); which can carry out 3D mapping by using 2D laser sensors; and which has a high accuracy level and can collect multipurpose data, was developed.

All the stages of this study were explained in main sections of this paper as Related Work, Description of the System and Conclusions.

2 Related Work

Robots need 3D data (X_O, Y_O, Z_O) of the robot's surroundings to map their surroundings in 3D, and 3D location information (X_L, Y_L, Z_L) of the robot to perform 3D positioning in a 3D area.

In their study, Hinzmann, et al. [2] obtained the odometry data of the robot in accordance with the depth information obtained by using image processing algorithms of the images obtained from the camera placed on flying robots for use in flying robots.

Teixeira, et al. [3] studied on determining the best route with predetermined start and end by using the SLAM algorithms known in the literature as well as telemetry data of Intel's AscTec Firefly drone.

In their study, Iacono and Sgorbissa [4] developed an obstacle avoidance algorithm from the data produced by an RGB-D camera (Microsoft Kinect) they installed on AscTec Firefly drone as an additional algorithm to the algorithms already used in the autonomous controlling of drones.

Kaufman, et al. [5], in their study, integrated variable motion planning algorithm to the Bayesian Probability Mapping algorithm in order to minimize the errors in Bayesian Probability Mapping algorithm that is used in the literature for mapping studies in 3D spaces of unknown shape. They also proposed that the 3D position can be used in the solution of the 3D problem by slicing it into 2D positioning method. Meanwhile, both mapping and reconnaissance algorithms were explained in this study through simulations and quadrotor flight trials.

In their study, Nguyen, et al. [6] developed and proposed a new method for creating a new curve path in case of an obstacle. They concluded that the method can create a safe path that takes into account obstacles detected in real time and prevents collisions.

Yu, et al. [7] performed 2D positioning and 3D mapping in a 3D environment with obstacles by using Rplidar and Kinect sensor with ROS operating system control.

In the study conducted by Yu, et al. [7], they created a composite coordinate positioning system by combining the indoor 2D map of the object created by the Gmapping algorithm and the 3D point cloud image information of the object. According to the result of the empirical studies conducted by Yu, et al. [7], they concluded that the positioning precision of the composite coordinate positioning system in this study is 6.7% higher than widespread ultrasonic and infrared positioning systems [8]; 20% higher than Bluetooth angle estimation positioning system [9]; and 72% higher than ultra-wideband positioning system [10].

Nellithimaru and Kantor [11] conducted a study concerning the classification of agricultural products using the SLAM algorithm they developed by adding a stereo camera. They stated that they achieved positive results thanks to this method in outdoor environments (without any pipeline used in such applications) without any constant lighting conditions or scene dynamics. They conducted this study in vineyards in order to count the crops of the farmers and estimate the yield of the crop to be obtained from the land.

When the studies in the literature are examined, it is seen that there are robots used for mapping from the ground and air. Ground and air robots that are used for mapping of indoor and outdoor spaces map their environments either by the control of an operator or by navigating autonomously [12] [13] [14] [15] [16] [17].

As for the terrestrial robots commonly used in the literature, it is seen that mapping reaches a high level of precision. However, when the physical conditions of the spaces such as mine pits, underground cities or caves where the study will be performed are examined, it is concluded that terrestrial robots are not suitable for these physical conditions due to fact that these robots must be used on flat or close-to-flat surfaces.

3 Description of System

In this study conducted, through the M100 Robot model created under Gazebo simulator, the control of the M100 Robot in real-world under the ROS environment is carried out by the developed nodes. The nodes developed, their relations with each other as well as the publisher and subscriber topics by these nodes are shown in Figure 1.

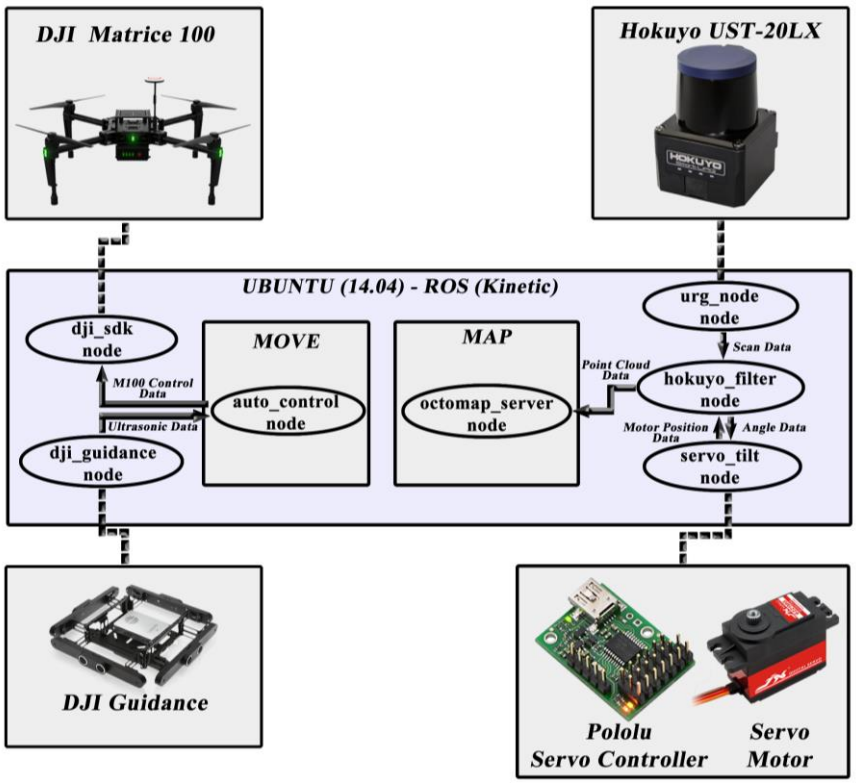

Figure 1
Block diagram of the study

3.1 Equipment

A DJI Guidance ultrasonic sensor kit, Hokuyo UST-20LX laser sensor, as well as, a Mini PC, running at 2.8 GHz processor (i5-6200U) with 4GB DDR3 Ram and 256 GB SSD hard disk, were placed on the M100 Robot (Figure 2).

Ubuntu 14.04 operating system was installed on the Mini PC due to the fact that the ROS Kinetic operating system used for the control of the M100 robot supports the Ubuntu 14.04 operating system. The control of M100 Robot and Hokuyo UST-20LX laser sensor is conducted by the ROS Kinetic operating system installed on the Mini PC.

Figure 2
3D Simultaneous Positioning and Mapping Robot in Dark, Indoor Environments with Autonomous Flying Robot

The Hokuyo UST-20LX laser sensor is capable of area scanning on the y-axis at an angle of 270° within a period of 25 ms. The resolution of this sensor is 0.25° (Equation 1) and the measuring range is 20-30 meters depending on the amount of light in the environment. In this study, the data transfer between Hokuyo UST-20LX laser sensor and Mini PC is carried out through RJ45 connection [18].

$$270° / 1080 \text{ (step)} = 0.25° \quad (1)$$

The servomotor driven electromechanical system in Figure 3 was designed so that the Hokuyo UST-20LX laser sensor can capture distance information by scanning its 3D environment in multi-axis. Thanks to this design, the Hokuyo UST-20LX sensor, which can scan in 2D, is equipped with 3D scanning capability.

Figure 3
Electromechanical system developed for 3D scanning of the Hokuyo UST-20LX sensor

A servo motor driver board in Figure 4 was used in order to control the movement of the Hoyuko UST-20LX sensor on the electro-mechanics system in Figure 3 between the 70° and 80° on Z axis through the Mini PC [19].

Figure 4
PC controlled, Pololu Micro Maestro 6-Channel USB Servo Controller

Through this moving system, the Hokuyo UST-20LX sensor taking measurements from horizontal axis was provided with the ability to make measurements from vertical axis (z axis) in addition to the horizontal axis.

3.2 Installing Hokuyo UST-20LX on the DJI Matrice 100 Flying Robot Platform

The Hokuyo UST-20LX laser sensor model of ROS's 'hukuyo_utm20lx' package was added to the x = 0, y = 0, z = 0.175 (mt) position of '*scanmatcher_frame*' frame where the M100 Robot that was modelled in Blender program is located through the XML codes in Figure 5.

```
<xacro:include filename="$(find  sensor_description)/urdf/hokuyo_utm20lx.urdf.xacro" />
<xacro:hokuyo_utm20lx name="laser0" parent="scanmatcher_frame" ros_topic="scan" update_rate="40"
                      ray_count="1081" min_angle="-270" max_angle="270">
  <origin xyz="0.0 0.0 0.175" rpy="0 0 0"/>
</xacro:hokuyo_utm20lx>
```

Figure 5
Installing Hokuyo UST-20LX on M100 Robot model

When the M100 Robot model as well as the Hokuyo UST-20LX model installed on the M100 Robot model are opened in the RViz visualizer, they appear on the RViz visualizer screen as in Figure 6.

Figure 6
Image of M100 Robot and M100 Robot model in RViz visualizer

The communication between the Hokuyo UST-20LX laser sensor and the PC is carried out via the TCP/IP protocol using the static IP address of 192.168.0.10.

3.3. Obtaining 3D Data from 2D Data taken from Hokuyo UST-20LX

The "create_point_cloud.launch" in Figure 7 is an XML file that was created for use in 3D mapping and that combines the nodes which are necessary to obtain 3D data from 2D data.

```
<launch>
  <node pkg="ros_pololu_servo" type="ros_pololu_servo_node" name="ros_pololu_servo_node" output="screen">
    <param name="pololu_motors_yaml" value="$(find hokuyo_spinner)/launch/pololu_motors.yaml" />
    <param name="port_name" value="/dev/ttyACM0" />
    <param name="baud_rate" value="115200" />
    <param name="rate_hz" value="5" />
    <param name="daisy_chain" value="false" />
  </node>
  <node name="servo_tilt" pkg="hokuyo_spinner" type="servo_tilt" output="screen" >
   <param name="max_angle_" type="int" value="80" />
   <param name="min_angle_" type="int" value="-70" />
   <param name="pause_time_" type="double" value="0.6" />
   <param name="motor_speed_" type="double" value="0.001" />
</node>
  <node pkg="urg_node" type="urg_node" name="urg_node">
   <param name="ip_address" value="192.168.0.10" />
   <param name="frame_id" value="laser0_frame" />
</node>
  <node name="hokuyo_filter" pkg="hokuyo_spinner" type="hokuyo_filter" output="screen" >
   <param name="min_distance_limit" type="double" value="0.5" />
   <param name="min_intensities_limit" type="double" value="0.0" />
  </node>
  <node name="servo_tilt_transform" pkg="hokuyo_spinner" type="servo_tilt_transform" output="screen" />
  <node type="laser_scan_assembler" pkg="laser_assembler" name="pcl_assembler_server">
   <remap from="scan" to="hokuyo_scan/m100_filtered/angle_scan"/>
   <param name="max_scans" type="int" value="400" />
   <param name="fixed_frame" type="string" value="/laser0_frame" />
  </node>
  <node type="pcl_assemblerr_client" pkg="hokuyo_spinner" name="pcl_assemblerr_client" output="screen">
     <param name="scan_time" type="double" value="5" />
   <param name="assembled_cloud_mode" type="string" value="subscriber" />
  </node>
</launch>
```

Figure 7

The "create_point_cloud.launch" XML file written for transition from 2D to 3D

While the '*ros_pololu_servo_node*' node in the "create_point_cloud.launch" file make the connection with the servo-motors of the drive board in ROS, the '*servo_tilt*' node ensures movement in Hokuyo UST-20LX on the vertical axis (The servo-motor is continuously moved between -70° and 80°). After the obstacle information taken from Hokuyo UST-20LX laser sensor by the '*urg_node*' node is filtered by the '*hokuyo_filter*' node, this information is shared in ROS environment in the topic form of '*/hokuyo_scan/m100_filtered/angle_scan*'. The 3D point cloud is created by combining the angle created by both '*pcl_assembler_server*' node and '*servo_tilt_transform'* (the tf angle of layers created in Figure 8) node as well as the topic of *'/hokuyo_scan/m100_filtered/angle_scan'*.

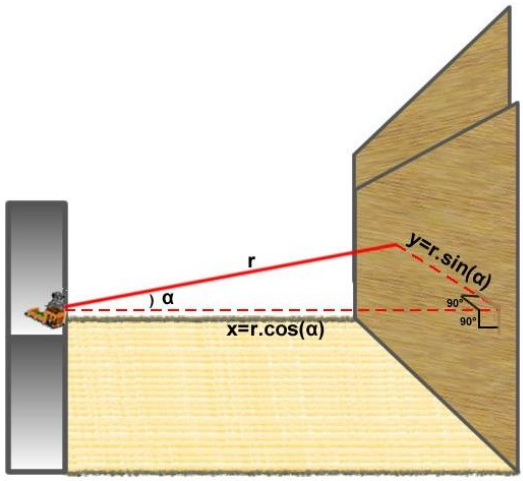

Figure 8
The 3D tf conversion of Hokuyo UST-20LX laser sensor

Equation 2 was used for 3D conversion of information from the Hokuyo UST-20LX laser sensor along the Y axis according to the view angle of M100 Robot.

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} r\,cos\alpha \\ r\sin\alpha \\ Z_{imu} \end{pmatrix} \tag{2}$$

Then, the 3D point cloud generated by the '*pcl_assembler_server*' node was shared in the topic form of '*/hokuyo_scan/m100_filtered/assembled_cloud*' in '*pcl_assemblerr_client*' node to be mapped in ROS environment.

Filtering data from Hokuyo UST-20LX: The Hokuyo UST-20LX laser sensor is capable of measuring within 270° along the Y-axis [18]. The data within the limits of 46°-225° were evaluated by filtering the measurement gaps of 0°-45° and 225°-270° due to the fact that the hardware of M100 robot runs into the view angle of Hokuyo UST-20LX laser sensor between 0°-270° measurement gap. In addition, during the movement of the electromechanical system in Figure 3 created to obtain 3D data from 2D data of the Hokuyo UST-20LX laser sensor, since some angle values on the z-axis correspond to the M100 Robot's equipment like wings, the measuring range was determined to be taken as 0.5 m and above. Thus, topic of '*/scan*' shared by '*urg_node*' in ROS is published in ROS environment being filtered by the '*hokuyo_filter*' node in order to be used with tf angular conversion as topic form of '*/hokuyo_scan/m100_filtered/angle_scan*'.

Angular tf conversion for transition from 2D to 3D: In order to perform the mapping in the RViz visualizer, the real-world M100 Robot and the M100 Robot model visualized in the RViz visualizer are programmed to work simultaneously with ROS. M100 Robot model and peripheral equipment models (links: laser

sensor, servo-motor mechanism, etc., which moves the laser sensor in the z axis) were placed on different layers in Rviz visualizer to be controlled by ROS in the visualizer. While the M100 Robot model is on the '*scanmatcher_frame*' layer, the Hokuyo UST-20LX laser sensor model is placed on the *'laser0_frame*' layer. Thus, the Hokuyo UST-20LX laser sensor information on the y-axis, which is received during the movement of the real-world Hokuyo UST-20LX laser sensor in the range of -70° to 80° on the z-axis, is converted into a 3D point cloud in ROS environment (Figure 9) by distributing the information to the '*laser0_frame*' layer that moves between -70° and 80° along the z-axis of the location, where the M100 Robot is positioned.

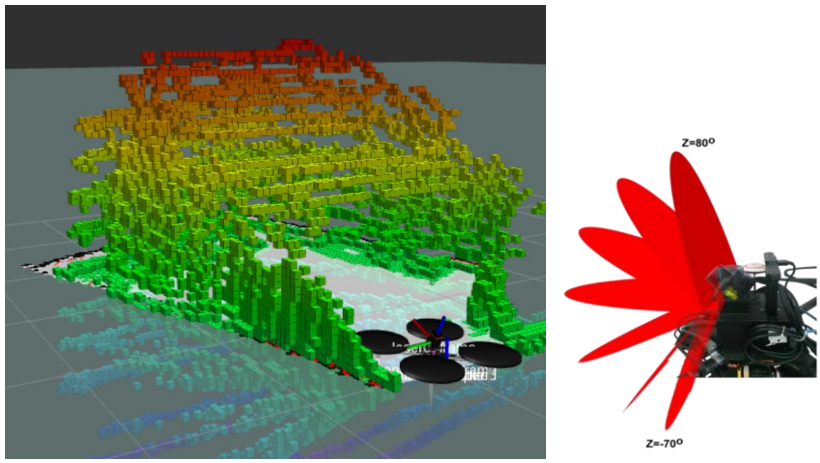

Figure 9
Conversion from 2D to 3D with Hokuyo UST-20LX laser sensor

3.4 The Control of M100 Robot

The autonomous control software for the autonomous movement of the M100 Robot is as in Figure 10. Autonomous movement of the M100 Robot was conducted by using obstacle information obtained from ultrasonic sensors.

The "obtain_control()" function in Figure 10 is a C ++ code block created to make the M100 Robot controllable through shared messages on ROS. Communication of the M100 Robot with the PC is carried out via the UART port of the M100 Robot. The M100 Robot's motors are activated by the "Arm_Control()" function.

The "m100_Going(string task, int speed)" function created for the autonomous control of the M100 Robot enables the movement of the M100 Robot in the direction of the '*task*' parameter that corresponds to the function as a parameter and at the speed of '*speed*' parameter that also corresponds to the function as a parameter.

```
#include <ros/ros.h>
#include <geometry_msgs/QuaternionStamped.h>
#include <geometry_msgs/Vector3Stamped.h>
#include <sensor_msgs/NavSatFix.h>
#include <std_msgs/UInt8.h>
#include <dji_sdk/DroneTaskControl.h>
#include <dji_sdk/SDKControlAuthority.h>
#include <dji_sdk/DroneArmControl.h>
#include <tf/tf.h>
#include <sensor_msgs/Joy.h>
#include <sensor_msgs/LaserScan.h>     //obstacle distance & ultrasonic
namespace DJI_m100{
   class Teleop{
      private:
         ros::NodeHandle nh0;
         ros::NodeHandle nh1;
         ros::ServiceClient sdk_ctrl_authority_service;
         ros::ServiceClient drone_task_service;
         ros::ServiceClient drone_arm_control_service;
         ros::Publisher Kontrolcu;
         ros::Subscriber ultrasonic_pub;
      public:
      Teleop(){
         sdk_ctrl_authority_service = nh0.serviceClient<dji_sdk::
SDKControlAuthority> ("dji_sdk/sdk_control_authority");
         drone_task_service        = nh0.serviceClient<dji_sdk::
DroneTaskControl>("dji_sdk/drone_task_control");
         drone_arm_control_service  = nh0.serviceClient<dji_sdk::
DroneArmControl>("dji_sdk/drone_arm_control");
         controller = nh1.advertise<sensor_msgs::
Joy>("/dji_sdk/flight_control_setpoint_ENUvelocity_yawrate", 10);
      }
      bool Arm_Control(){
         dji_sdk::DroneArmControl Arm_Motor_Control;
         Arm_Motor_Control.request.arm=1;
         drone_arm_control_service.call(Arm_Motor_Control);
         if(!Arm_Motor_Control.response.result) {
            ROS_ERROR("MOTOR CONTROL IS FAILED!");
            return false;
         }
         ROS_INFO("MOTOR CONTROL IS OK");
         return true;
      }
      bool obtain_control(){
         dji_sdk::SDKControlAuthority authority;
         authority.request.control_enable=1;
         sdk_ctrl_authority_service.call(authority);
         if(!authority.response.result) {
            ROS_ERROR("CONTROL IS FAILED!");
            return false;
         }
         ROS_INFO("CONTROL IS OK");
         return true;
      }
      bool takeoff_land(int task) {
         dji_sdk::DroneTaskControl droneTaskControl;
         droneTaskControl.request.task = task;
         drone_task_service.call(droneTaskControl);
         if(!droneTaskControl.response.result) {
            ROS_ERROR("PROCCESS IS FAILED -1");
            return false;
         }
         else{
            ROS_INFO("PROCESS IS OK :)");
         }
         return true;
      }
      void UltrasonicCallback(const sensor_msgs::Ultrasonic::ConstPtr&
ultrasonic) {
        if(5 > ultrasonic->ultrasonic[0] * 0.001f && 5> ultrasonic-
>ultrasonic[1] * 0.001f) {
          if(ultrasonic->ultrasonic[0] * 0.001f > ultrasonic->ultrasonic[1] *
0.001f) {
            while((ultrasonic->ultrasonic[0] * 0.001f > ultrasonic-
>ultrasonic[1] * 0.001f))
               m100_Going("Top", 1);
            }
            else  if(ultrasonic->ultrasonic[0] * 0.001f < ultrasonic-
>ultrasonic[1] * 0.001f) {
              while(ultrasonic->ultrasonic[0] * 0.001f < ultrasonic-
>ultrasonic[1] * 0.001f)
                 m100_Going("Down", 1);
            }
        }
        else{
          while(ultrasonic->ultrasonic[1] * 0.001f !=5)
            m100_Going("Top", 1);
        }
        if(ultrasonic->ultrasonic[2] * 0.001f > ultrasonic->ultrasonic[4] *
0.001f) {
          while(ultrasonic->ultrasonic[2] * 0.001f > ultrasonic->ultrasonic[4]
* 0.001f)
            m100_Going("Right", 1);
        }
        else  if(ultrasonic->ultrasonic[2] * 0.001f < ultrasonic->ultrasonic[4]
* 0.001f) {
          while(ultrasonic->ultrasonic[2] * 0.001f < ultrasonic->ultrasonic[4]
* 0.001f)
            m100_Going("Left", 1);
        }
        if(0 > ultrasonic->ultrasonic[3] * 0.001f) {
          while(0 > ultrasonic->ultrasonic[3] * 0.001f)
            m100_Going("Front", 1);
        }
       }
       bool m100_Going(string task, int speed) {
          obtain_control();
          Arm_Control();
          sensor_msgs::Joy controlPosYaw;
          float Xvel=0,Yvel=0,Zvel=0,Rvel=0;
          if(task=="right") {
             Yvel=speed; //FOR Y coordinate
          }
          else if(task=="left"){
             Yvel=-1*speed;
          }
          if(task=="front") {
             Xvel=speed; //FOR X coordinate
          }
          else if(task=="back"){
             Xvel=-1*speed;
          }
          if(task=="Top")  {
             Zvel=1*speed; //FOR Z coordinate
          }
          else if(task=="Down"){
             Zvel=-1*speed;
          }
          if(task=="R90") {
             Rvel=$pi*speed*(-1); //FOR Radial Rotate
          }
          else if(task=="L90"){
             Rvel=$pi*speed*(1);
          }
          controlPosYaw.axes.push_back(Yvel);
          controlPosYaw.axes.push_back(Xvel);
          controlPosYaw.axes.push_back(Zvel);
          controlPosYaw.axes.push_back(Rvel);
          Kontrolcu.publish(controlPosYaw);
       }
    };
}
int main(int argc, char** argv) {
  ros::init(argc, argv, "m100_control");
  DJI_m100::Teleop teleop;
  ros::NodeHandle n;
  ultrasonic_pub = n.subscribe("/guidance/ultrasonic",10,
UltrasonicCallback);
  return 0;
}
```

Figure 10
The autonomous control software of M100 Robot

A repeating "UltrasonicCallback()" function at 10 Hz was created (The reading frequency of the DJI Guidance sensor from the ultrasonic sensors is 10 Hz [20]) for the control of the ultrasonic sensor located on the right, left, front, under and top of the M100 Robot and for calling the "M100_going()" function with direction and speed parameters in accordance with the distance information coming from the sensors.

The autonomous control software for M100 Robot in Figure 10 was created with possible scenarios in which the M100 Robot may encounter in closed spaces. Among these scenarios are;

- Raising and lowering of M100 Robot in order to balance the top-bottom distance if the distance information from the ultrasonic sensors on the top and bottom of the M100 Robot is less than 5 m,
- Keeping the altitude of the M100 Robot at 5 m and keeping this altitude if the distance information from the ultrasonic sensors on the top and bottom of the M100 Robot is not more than 5 m,
- Ensuring the balance of M100 Robot by moving to the right if the information from the ultrasonic sensors from the right side are less from the left side of the M100 Robot or to the left direction for a vice versa situation is valid,
- The advancement of the M100 Robot if there is no obstacle ahead, according to the distance information coming from the ultrasonic sensors.

The coordinate axes in Figure 11 are used for X, Y, Z direction movement of M100 Robot.

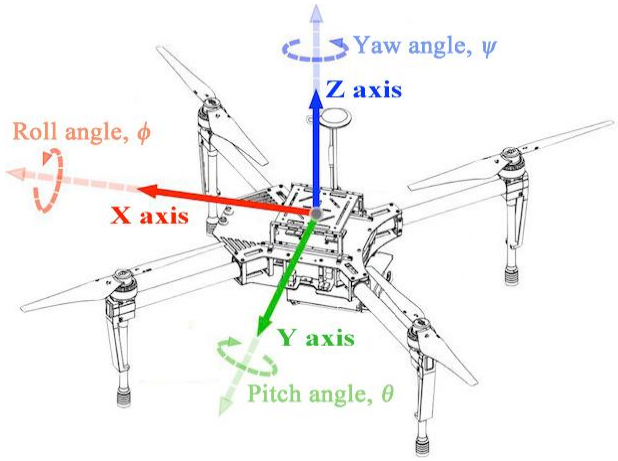

Figure 11
Coordinate axes of the M100 Robot

4 Practical Applications

The mapping studies were carried out in caves in Turkey, Isparta, Atabey district at coordinates of 37°53'41.8"N 30°32'58.5"E and 37°53'39.0"N 30°32'42.3"E.

1. Mapping Study: The entrance of the cave at the coordinates of 37°53'41.8"N 30°32'58.5"E is as in Figure 12.

Figure 12
The entrance of the cave at the coordinates of 37°53'41.8"N 30°32'58.5"E

This cave in Figure 12 opens to 400-500 m inwards from the foot of the mountain. The height of the cave provides a space where a person of 175 cm height can walk upright (Figure 13).

Figure 13
Inside of the cave at the coordinates of 37°53'41.8"N 30°32'58.5"E

The interior of the cave in Figure 13 is very dark and dangerous. In this study, carried out to map such dark and closed spaces, the map of this cave, at the coordinates of 37°53'41.8"N 30°32'58.5"E, was created, as shown in Figure 14.

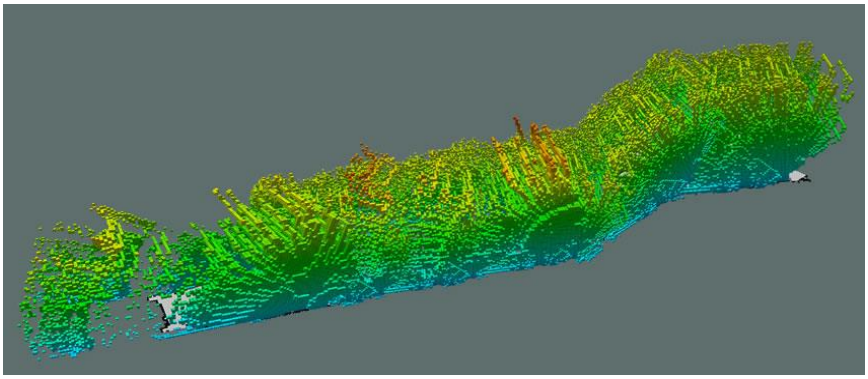

Figure 14
The map of the cave at the coordinates of 37°53'41.8"N 30°32'58.5"E

The M100 Robot moved autonomously starting from the entrance of the cave. Since the M100 Robot moves slowly during the rotation around its own axis and due to its slow movement during direction changes on z-axis (movement in the -z and +z direction), as determined in the control software of the robot, there are some accumulations on some point clouds as seen in Figure 14. At the last stage of the mapping process, due to the fact that the right and left side of the M100 Robot was closed, and thus, had to make a 180° rotation on its axis, there are some missing point clouds, at the last section of the map in Figure 14. Yet, it can be seen that a map identical to the cave could be obtained when the map was studied in general terms.

2. Mapping Study: The entrance of the cave at the coordinates of 37°53'39.0"N 30°32'42.3"E is as in Figure 15.

Figure 15
The entrance of the cave at the coordinates of 37°53'39.0"N 30°32'42.3"E

The location of this cave in Figure 15, is approximately 100 m above the foot of the mountain. It has an area extending to the right, after the entrance gate of the cave. The interior of the cave has a rectangular prism structure as in Figure 16.

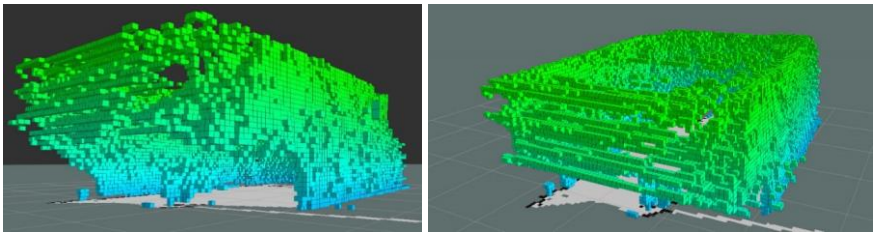

Figure 16
The map of the cave at the coordinates of 37°53'39.0"N 30°32'42.3"E

When the map in Figure 16 was studied, it was seen that a map identical to the cave at the coordinates of 37°53'39.0"N 30°32'42.3"E could be obtained.

Conclusions

In this study, an autonomous, high-precision, fast and low-cost system that has the ability to map in 3D was created. Dark, rugged and dangerous closed spaces, such as caves, underground cities and mines were mapped in 3D, with this system. The system has no limit on movement (bumpy, hollowed or sloping spaces).

In order to obtain the obstacle information, during autonomous movement, the developed system used within the scope of this study, 5 ultrasonic sensors – right, left, front, top, bottom – were used. For the purpose of data collection from Hokuyo UST-20LX laser sensor that can collect 2D data (X and Y axis) along the Z axis, a servomotor driven electromechanical equipment was developed so that the developed system can map its environment in 3D during its autonomous movement. The control of the hardware developed and used in this study was carried out by ROS nodes written in C++ programming language.

The developed system can not only be actively used in mining operations, where changes must be continuously monitored, especially these days when occupational health and safety are at the forefront, but also in the discovery of underground cities and caves, which are important sources of economy and tourism in countries, like Turkey. The system can also provide literature contributions to National and International Academic Studies.

Acknowledgments

This research was brought to life through the Project 116E013, which is supported by the Scientific and Technical Research Council of Turkey (TÜBİTAK). We would like to thank them for their contributions.

References

[1] Soma_Mine_Disaster (2014, 30.12.2015) Soma Mine Disaster [Online]. Available: https://en.wikipedia.org/wiki/Soma_mine_disaster

[2] T. Hinzmann, J. L. Schönberger, M. Pollefeys, and R. Siegwart, "Mapping on the fly: Real-time 3D dense reconstruction, digital surface map and incremental orthomosaic generation for unmanned aerial vehicles," in Field and Service Robotics, 2018: Springer, Cham, pp. 383-396

[3] L. Teixeira, I. Alzugaray, and M. Chli, "Autonomous aerial inspection using visual-inertial robust localization and mapping," in Field and Service Robotics, 2018: Springer, Cham, pp. 191-204

[4] M. Iacono and A. Sgorbissa, "Path following and obstacle avoidance for an autonomous UAV using a depth camera," Robotics and Autonomous Systems, Vol. 106, pp. 38-46, 2018

[5] E. Kaufman, K. Takami, Z. Ai, and T. Lee, "Autonomous Quadrotor 3D Mapping and Exploration Using Exact Occupancy Probabilities," in 2018 Second IEEE International Conference on Robotic Computing (IRC), 2018: IEEE, pp. 49-55

[6] P. D. Nguyen, C. T. Recchiuto, and A. Sgorbissa, "Real-time path generation and obstacle avoidance for multirotors: a novel approach," Journal of Intelligent & Robotic Systems, Vol. 89, No. 1-2, pp. 27-49, 2018

[7] Y. Yu, Y. Piao, Y. Ni, and T. Si, "Research on Accurate Positioning of Indoor Objects Based on ROS and 3D Point Cloud," in Journal of Physics: Conference Series, 2019, Vol. 1229, No. 1: IOP Publishing, p. 012005

[8] F. Yao, C. Tao, and S. Li, "Indoor positioning system research and implementation based on phase of arrival," Electronic Measurement Technology, Vol. 39, No. 11, pp. 30-35, 2016

[9] M. W. Liu, T. J. Liu, Y. Ye, and L. Wu, "The Application Research of Indoor Positioning Based on Low-power Bluetooth Technology," Wireless Communication Technology, Vol. 24, No. 3, pp. 19-23, 2015

[10] J. Zheng, H. Qian, and L. Wang, "An improved DV-Hop positioning algorithm for wireless sensor network," in 2015 IEEE International Conference on Progress in Informatics and Computing (PIC), Nanjing, China, 2015, Nanjing, China: IEEE

[11] A. K. Nellithimaru and G. A. Kantor, "ROLS: Robust Object-Level SLAM for Grape Counting," in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, California, 2019: IEEE Xplore, p. 9

[12] I. Peitzsch, B. Liboy, S. Biaz, and R. Chapman, "3D Mapping Using a UAV, an IMU, and a 2D LiDAR," 2019

[13] A. Molnár, "Surveying Archaeological Sites and Architectural Monuments with Aerial Drone Photos," Acta Polytechnica Hungarica, Vol. 16, No. 7, 2019

[14] B. Guo, H. Dai, Z. Li, and W. Huang, "Efficient Planar Surface-Based 3D Mapping Method for Mobile Robots Using Stereo Vision," IEEE Access, Vol. 7, pp. 73593-73601, 2019

[15] T. Pozderac, J. Velagić, and D. Osmanković, "3D Mapping Based on Fusion of 2D Laser and IMU Data Acquired by Unmanned Aerial Vehicle," in 2019 6th International Conference on Control, Decision and Information Technologies (CoDIT), 2019: IEEE, pp. 1533-1538

[16] D. Backes, G. Schumann, F. Teferele, and J. Boehm, "Towards a high-resolution drone-based 3D mapping dataset to optimise flood hazard modelling," The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. 42, No. W13, pp. 181-187, 2019

[17] Z. Jiang, J. Zhu, Z. Lin, Z. Li, and R. Guo, "3D mapping of outdoor environments by scan matching and motion averaging," Neurocomputing, 2019

[18] UST-20LX (2014, 20.11.2016) Distance Data Output/UST-10/20LX [Online] Available: https://www.hokuyo-aut.jp/search/single.php?%20serial=167

[19] M. Maestro. Pololu Maestro Servo Controller User's Guide [Online] Available: https://www.pololu.com/docs/pdf/0J40/maestro.pdf

[20] Guidance_User_Manual_EN. DJI Guidance User Manual [Online] Available: http://download.dji-innovations.com/downloads/dev/Guidance/en/Guidance_User_Manual_en_V1.6.pdf