src/drive/include/TB3Drive.h

```
1  // Original class was turtlebot_drive taken from simulation example package
2  // Class TB3Drive is the main class that controls the bot based on
3  // lidar and odom data published by CLidar and CPose nodes.
4
5  #ifndef TB3DRIVE_H_
6  #define TB3DRIVE_H_
7
8  #include <ros/ros.h>
9
10 #include <sensor_msgs/LaserScan.h>
11 #include <geometry_msgs/Twist.h>
12 #include <nav_msgs/Odometry.h>
13 #include <vector>
14 #include <std_msgs/Float64MultiArray.h>
15 #include <std_msgs/Float64.h>
16
17
18 // Lidar indexing
19 const int CENTER = 0;
20 const int LEFT   = 1;
21 const int RIGHT  =  2;
22
23 // Velocity
24 const double STOP_FOWARD_V = 0.0;
25
26 // Bot states
27 const int STRAIGHT= 0;
28 const int LEFT_TURN = 1;
29 const int CORNER_TURN   = 2;
30 const int DEFAULT_STATE = 3;
31
32
33 // TB3DRive interface-------------------------------------------------------
34 // This class controls the robot based on the lidar readings. The class has the
35 // capabilities to transit states of the robot and compute linear and angular
36 // velocities and publish to cmd_vel to set velocity of the bot.
37
38 class TB3Drive
39 {
40  public:
41   TB3Drive();
42   ~TB3Drive();
43   bool controlLoop();
44
45  private:
46   // ROS NodeHandle
47   ros::NodeHandle nh_;
48   ros::NodeHandle nh_priv_;
49
50   // ROS Topic Publishers
51   ros::Publisher cmd_vel_pub_;
52
53   // ROS Topic Subscribers
54   ros::Subscriber cLidarSub;
55   ros::Subscriber cBotSub;
56
```

```
57    double forwardTarget;
58    double forwardTargetTurn;
59    double sideTarget;
60
61    double maxTurnVel;
62
63    double maxForwardVel;
64    double minForwardVel;
65
66    double angularVel;
67    double linearVel;
68
69    double turnKp;               // Proportional gain for angular velocity
70    double forwardKp;            // Proportional gain for linear velocity
71
72    double tb3Pose;             // Current Position form odometry - sent to by
    CPose
73    double prevTB3pose;         // Previous Position from odometry
74
75    int leftTurnFlag;
76
77    std::vector<double>lidarData;
78
79    // Function publishes to cmd_vel topic to control linear
80    // and angular velocity of turtlebot.
81    void updatecommandVelocity(double linear, double angular);
82
83    // Callback functions receiving messages from CPose class and CLidar class
84    void cLidarMsgCallBack(const std_msgs::Float64MultiArray::ConstPtr &msg);
85    void cPoseMsgCallBack(const std_msgs::Float64::ConstPtr &msg);
86 };
87 #endif
88
```