

```

/**
 * @file autoRight.c
 * @brief Right side autonomous routines
 * Copyright (C) 2017 Ethan Wells
 *
 * This program is free software: you can redistribute it and/or modify it
 * under the terms of the GNU General Public License as published by the Free
 * Software Foundation, either version 3 of the License, or (at your option) any
 * later version.
 *
 * This program is distributed in the hope that it will be useful, but WITHOUT
 * ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS
 * FOR A PARTICULAR PURPOSE. See the GNU General Public License for more
 * details.
 *
 * You should have received a copy of the GNU General Public License along
 * with this program. If not, see <https://www.gnu.org/licenses/>
 */

#include "../include/auto.h"

void autonRight12() {
    getMogo(); // Get the mobile goal

    turnTo(5, 300); // Align to a right tilt
    driveSettings[0].max -= 40; // Limit left side speed
    driveToPosition(300, 700, 2200); // Back up
    driveSettings[0].max += 40; // Correct speed
    turnTo(165, 2000); // Turn around
    delay(400);

    // Reset drive encoders & gyro
    sensorReset(drive[0].sensor);
    sensorReset(drive[1].sensor);
    sensorReset(&gyro);

    driveToPositionAngle(900, 1100, -13, 1850); // Drive arc -13 degrees
    GO(placeConeT, NULL); // Place cone
    mogoP(MOGO_DOWN);

    driveSet(-127, -127); // Back up the drive
    delay(130);
    mogoP(MOGO_DOWN - 300); // Bring the mobile goal up a bit
    delay(250);
    driveSet(0, 0); // Stop the drive
}

```

```

        armSettings.target = arm.sensor->averageVal; // Reset the arm position to it's
                                                    // current position
    } /* autonRight12 */

void autonRight22() {
    getMogo(); // Get the mobile goal

    gyroSettings[0].tolerance--;
    gyroSettings[1].tolerance--;
    turnTo(10, 575); // Align to a right tilt
    gyroSettings[0].tolerance++;
    gyroSettings[1].tolerance++;
    driveSettings[0].max -= 40; // Limit left side speed
    GO(placeConeT, NULL); // Place cone
    driveToPosition(788, 388, 2200); // Back up
    driveSettings[0].max += 40; // Correct speed
    turnTo(-158, 2000); // Turn around

    // Reset drive encoders & gyro
    sensorReset(drive[0].sensor);
    sensorReset(drive[1].sensor);
    sensorReset(&gyro);

    driveToPositionAngle(1300, 1400, -13, 1800); // Drive arc -13 degrees

    // turnTo(59, 750);

    sensorReset(drive[0].sensor);
    sensorReset(drive[1].sensor);
    sensorReset(&gyro);

    driveToPositionAngle(1050, 1050, 0, 1425); // Drive straight

    driveSet(30, 30);
    mogoP(MOGO_DOWN); // Drop mobile goal

    // Wait a bit for the mobile goal to settle
    driveSet(100, 100);
    delay(350);

    driveSet(-127, -127); // Back up the drive
    delay(650);
    TaskHandle mogoUpHandle = GO(mogoPT, MOGO_UP);
    delay(500); // Make sure that the robot isn't touching a field
                // element
    driveSet(0, 0); // stop the robot

```

```
while (taskGetState(mogoUpHandle) != TASK_DEAD) {  
    delay(10);  
}  
  
armSettings.target = arm.sensor->averageVal; // Reset the arm position to it's  
                                              // current position  
} /* autonRight22 */
```