Tele-Op Documentation

**\*\*\*NEED TO CHANGE PAGE & SECTION NUMBERS\*\*\*DELETE WHEN DONE\*\*\***

This documentation is split up into four parts on the robot that need to be controlled: the wheels, arm, claw, and foundation movers. Each part consists of different versions of the code for that part, each with a brief description.

**Chassis: Mecanum wheels**

Basic Mecanum Wheel Program

**double** speed = -**gamepad1**.**left\_stick\_y**;  
**double** strafe = **gamepad1**.**left\_stick\_x**;  
**double** turn = -**gamepad1**.**right\_stick\_x**;  
  
  
**robot**.**leftFrontMotor**.setPower(speed + strafe + turn);  
**robot**.**rightFrontMotor**.setPower(speed - strafe - turn);  
**robot**.**rightBackMotor**.setPower(speed + strafe - turn);  
**robot**.**leftBackMotor**.setPower(speed - strafe + turn);

Slow mode is needed for accurate positioning of the robot to pick up and place a stone:

**if** (**gamepad1**.**a**) **speedMultiplier** = 0.5;  
**if** (**gamepad1**.**b**) **speedMultiplier** = 1.0;

**double** speed = (-**gamepad1**.**left\_stick\_y** \* **speedMultiplier**);

**double** strafe = **gamepad1**.**left\_stick\_x** \* **speedMultiplier**;  
**double** turn = -**gamepad1**.**right\_stick\_x** \* **speedMultiplier**;

The Arm driver needs to also be able to control the robot going forward and backward for precise positioning when picking up stones.

**double** speed = (-**gamepad1**.**left\_stick\_y** \* **speedMultiplier**) + (-**gamepad2**.**left\_stick\_y** \* 0.3);

**Arm: software for the stone scoring assembly**.

The arm moves in polar coordinates, because the arm on the robot rotates around an axis.

How:

**int** positionChangeAngle = (**int**)(-**gamepad2**.**right\_stick\_y** \* 6);  
**currentPositionAngle** += positionChangeAngle;  
**if**(**currentPositionAngle** > 1400) **currentPositionAngle** = 1400;  
**if**(**currentPositionAngle** < 0) **currentPositionAngle** = 0;

**int** positionChangeExtension = (**int**)((**gamepad2**.**right\_trigger** - **gamepad2**.**left\_trigger**) \* 10);  
**currentPositionExtension** += positionChangeExtension;

**if**(**currentPositionExtension** > 1730 ) **currentPositionExtension** = 1730;  
**if**(**currentPositionExtension** < 0) **currentPositionExtension** = 0;  
  
**extensionMotor**.setTargetPosition(**currentPositionExtension**);

Test: There was a problem where whenever you raised the arm, the extension would go backward, and when you lowered the arm, the extension would go forward. This is because the extension sprocket is concentric with the arm pivot axis.

Improve: We had to set an offset that increased as the arm was raised, so that the extension would stay at the same place.

**double extensionMotorAngleFactor** = E***XTENSION\_MOTOR\_COUNTS\_PER\_REV***/***ANGLE\_MOTOR\_COUNTS\_PER\_REV***;

**int** extensionPositionOffset = (**int**)((**double**)**currentPositionAngle**\***extensionMotorAngleFactor**);  
**int** positionChangeExtension = (**int**)((**gamepad2**.**right\_trigger** - **gamepad2**.**left\_trigger**) \* 10);  
**currentPositionExtension** += positionChangeExtension;  
**telemetry**.addData(**"Extension "**,**currentPositionExtension**);  
  
**if**(**currentPositionExtension** > 1730 ) **currentPositionExtension** = 1730;  
**if**(**currentPositionExtension** < 0) **currentPositionExtension** = 0;  
  
**extensionMotor**.setTargetPosition(**currentPositionExtension** + extensionPositionOffset);

What: Changing arm coordinates from polar coordinates to xy coordinates. Our drivers had to guess when to start moving the arm down, but with xy coordinates, they could just move the block directly above the skyscraper and then stack the block.

How:

**public boolean** moveArmXY(**double** x, **double** y) {  
 **double** dTheta = 2.5/Math.*hypot*(x,y);  
 **return** moveArm(Math.*toDegrees*(Math.*atan2*(y, x)+dTheta), Math.*hypot*(x, y));  
  
}

Test:

It works, but we have to start in the exact same place every time for the arm grabber to go straight up and down.

Improve: We don’t want to have to set the arm in the exact same position for teleop, so we made a seperate program to reset the motor encoders.

**public** DcMotor **angleMotor**;  
**public** DcMotor **extensionMotor**;  
  
**public void** init(){  
 **angleMotor** = **hardwareMap**.**dcMotor**.get(**"angleMotor"**);  
 **angleMotor**.setMode(DcMotor.RunMode.***STOP\_AND\_RESET\_ENCODER***);  
 **extensionMotor** = **hardwareMap**.**dcMotor**.get(**"extensionMotor"**);  
 **extensionMotor**.setMode(DcMotor.RunMode.***STOP\_AND\_RESET\_ENCODER***);  
}

**Grabber: Claw software for the stone scoring assembly.**

There are servos for grabbing and rotating the claw in two directions. The claw should always be perpendicular to the ground. The rotation servo could move 90 degrees at a time or we could move it with a joystick.

How:

**double verticalServoAngleFactor** = 1.333/***ANGLE\_MOTOR\_COUNTS\_PER\_REV***;

**double** verticalAngleOffset = (**double**)**currentPositionAngle**\***verticalServoAngleFactor**;  
**double** verticalServoPosition = 0.55 + verticalAngleOffset;

**verticalServo**.setPosition(verticalServoPosition);

**if**(**gamepad2**.**a**) { *//test positions TBD* **grabberServo**.setPosition(1.0);  
 **grabberServoTwo**.setPosition(0.25);  
}  
**if**(**gamepad2**.**b**) {  
 **grabberServo**.setPosition(0.5);  
 **grabberServoTwo**.setPosition(0.75);  
}

**if**(**gamepad2**.**dpad\_right**){  
 **rotationServo**.setPosition(0.18);

}  
  
**if**(**gamepad2**.**dpad\_up**){  
 **rotationServo**.setPosition(0.5);

}  
  
**if**(**gamepad2**.**dpad\_left**){  
 **rotationServo**.setPosition(0.82);

}

**double** currentPositionRotation = **rotationServo**.getPosition();  
**int** positionChangeRotation = (**int**) (**gamepad2**.**left\_stick\_x** \* 0.01);  
**rotationServo**.setPosition(currentPositionRotation + positionChangeRotation);

Test: When we rotate the claw, it would hit the arm and then go past it.

Improvement, we move the rotation axis perpendicular to the arm before rotating the block, then move the block back to level when we’re done with the rotation:

**if** (**timer**.seconds() > 0.5 && **timer**.seconds() < 1) {  
 **if** (**rotationDirection** == **"up"**) **robot**.**rotationServo**.setPosition(0.47);  
 **if** (**rotationDirection** == **"left"**) **robot**.**rotationServo**.setPosition(0.82);  
 **if** (**rotationDirection** == **"right"**) **robot**.**rotationServo**.setPosition(0.11);  
}  
**if** (**timer**.seconds() > 1) {  
 **robot**.**verticalServo**.setPosition(verticalServoPosition);  
}  
  
**if**(**gamepad2**.**dpad\_right**) {  
 **robot**.**verticalServo**.setPosition(0.5);  
 **timer**.reset();  
 **rotationDirection** = **"right"**;  
}  
  
**if**(**gamepad2**.**dpad\_left**) {  
 **robot**.**verticalServo**.setPosition(0.5);  
 **timer**.reset();  
 **rotationDirection** = **"left"**;  
}  
  
**if**(**gamepad2**.**dpad\_up**){  
 **robot**.**verticalServo**.setPosition(0.5);  
 **timer**.reset();  
 **rotationDirection** = **"up"**;  
}

**Foundation movers: To move the foundation is and out of the building zone.**

How:

**if** (**gamepad1**.**x**) r**obot**.**foundationServo**.setPosition(**0.82**);  
  
**if** (**gamepad1**.**y**) **robot**.**foundationServo**.setPosition(**0.15**);

Test:

It works, and we can move a skyscraper 6 stones high without it falling.