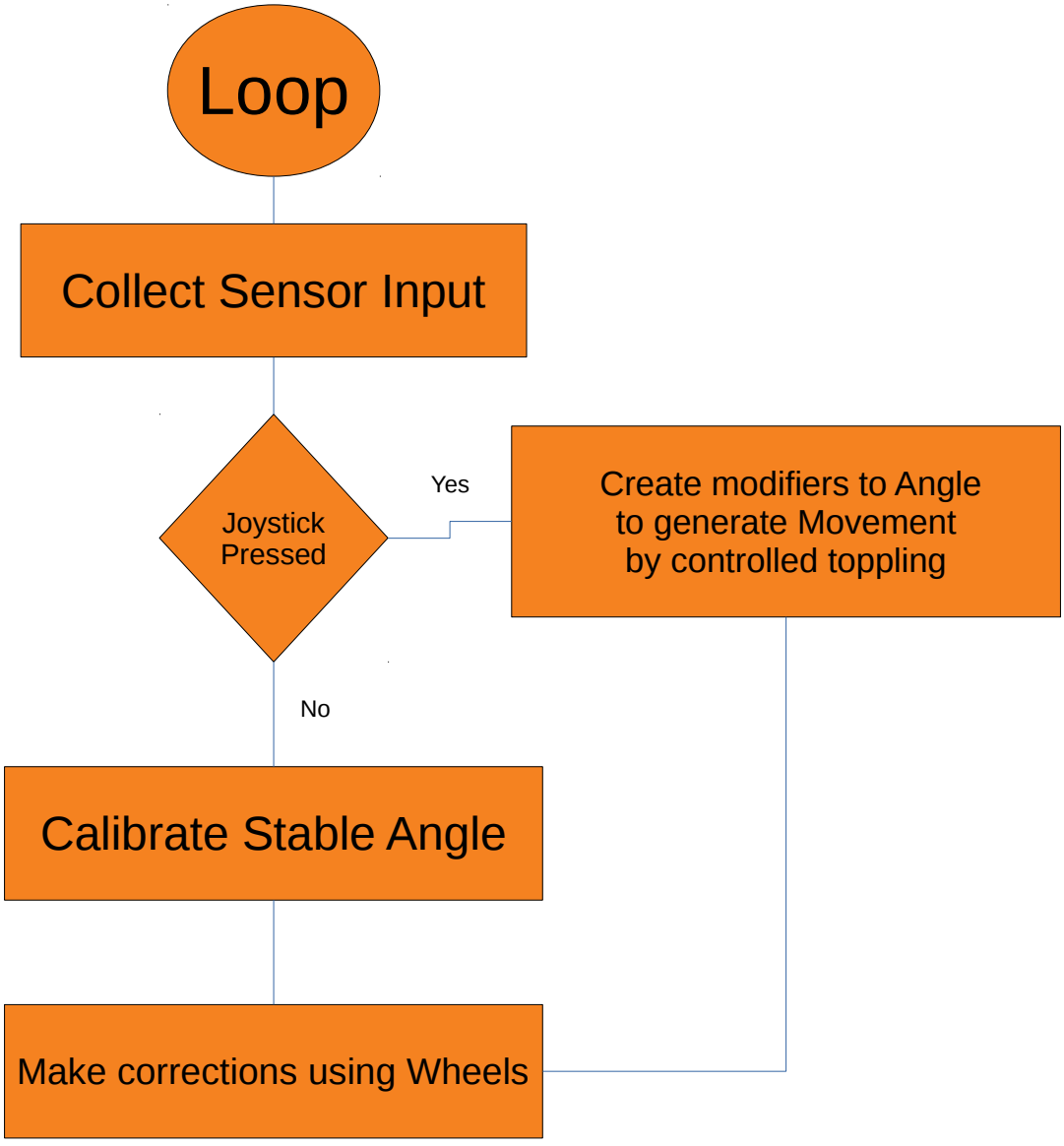
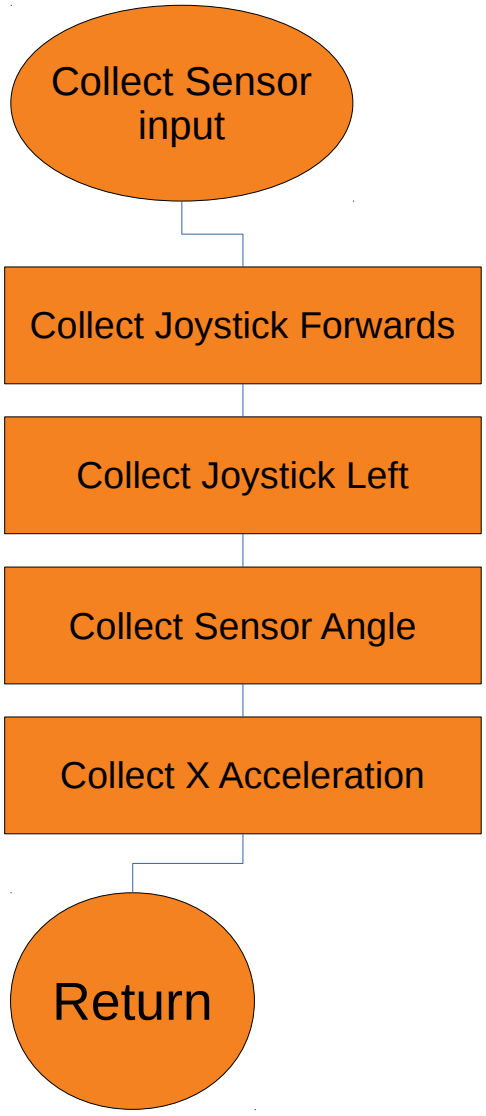


Sample Code:

```
void loop()
{
	collectSensorInput();

	if ( joystickPressed() )
	{
		createMovementPatchAngle();
	}
	else
	{
		calibrateStableAngle();
	}

	correctUsingWheels();
}
```



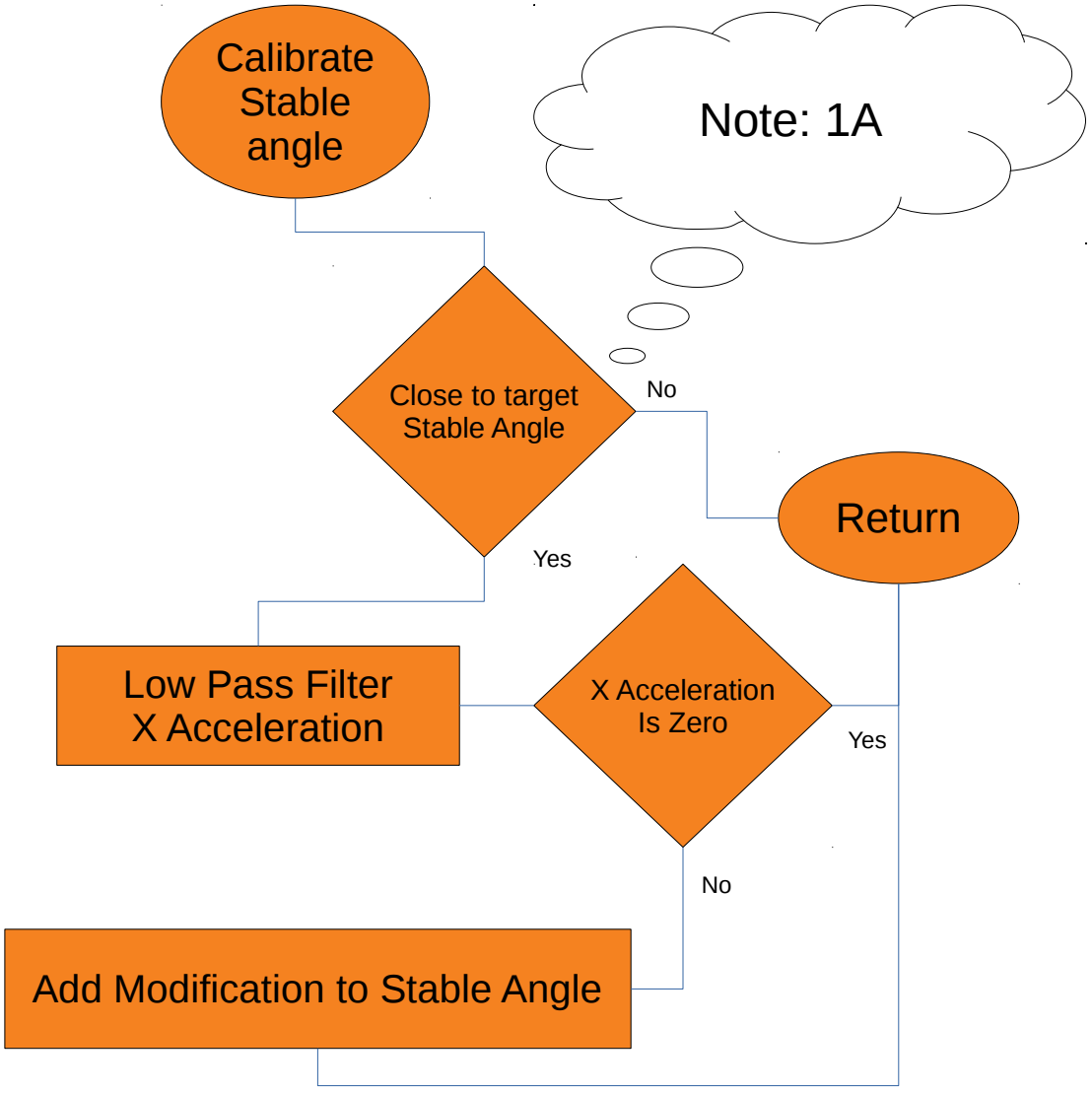
Sample Code:

```
void collectSensorInput()
{
	/* Store the sensor input in a global struct */

	joystickForwards = map(
		analogRead(JOYSTICK_FORWARDS_PIN),
		0, 255, -127, 128
	);

	joystickLeft = map(
		analogRead(JOYSTICK_LEFT_PIN),
		0, 255, -127, 128
	);

	…
}
```

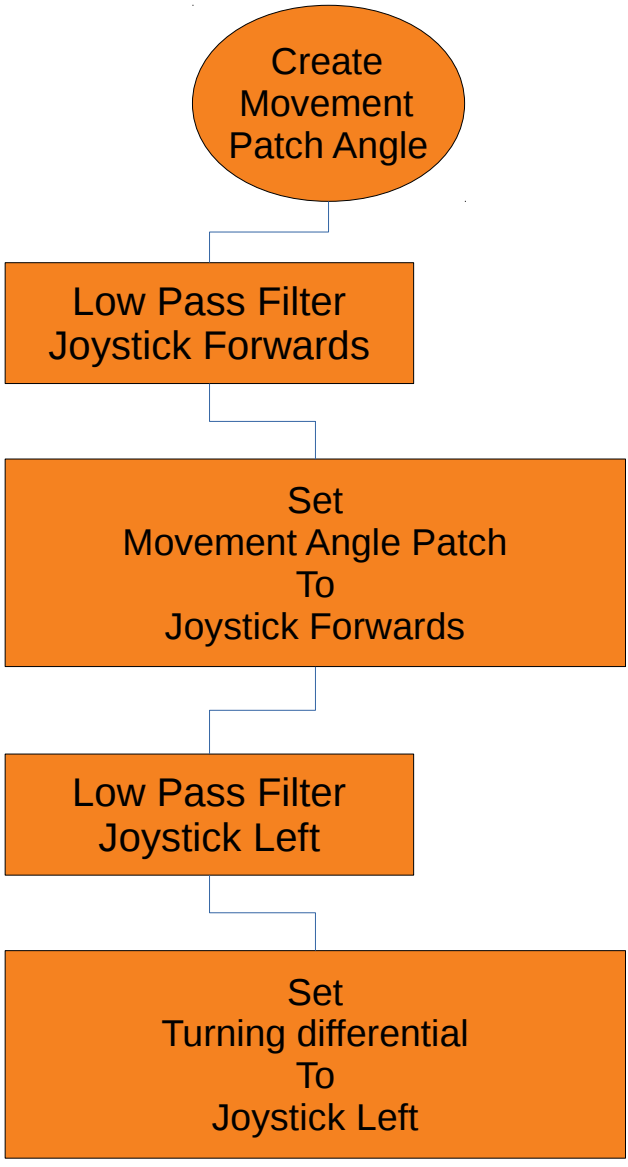


Sample Code:

```
void calibrateStableAngle()
{
	if( abs(sensorAngle - targetStableAngle) < something) )
	{
		if(XAcceleration > something)
		{
			calibratedStableAngle += (XAcceleration * something);
		}
	}
}
```

Notes:

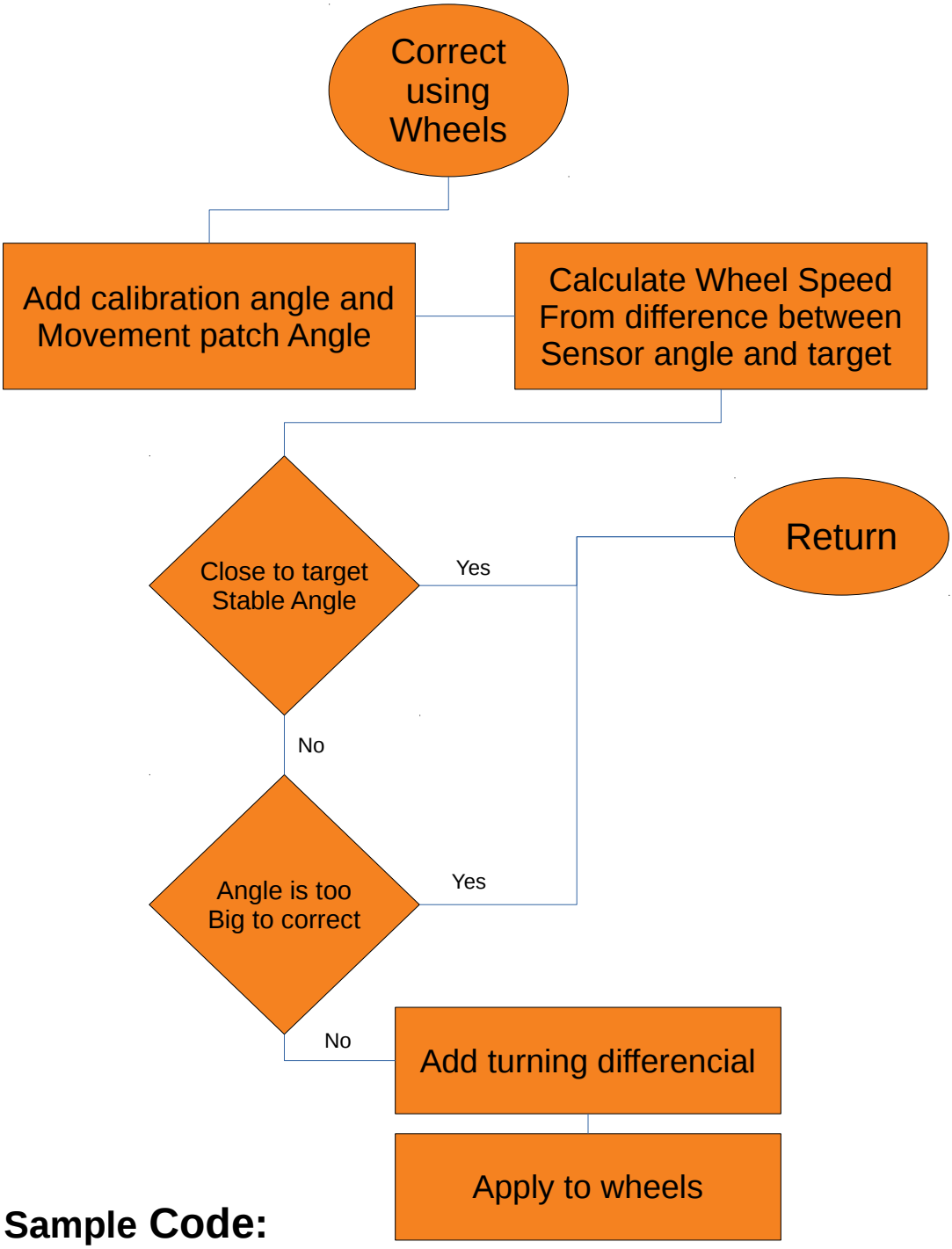
1A: If the Segway is moving towards a new target center, after the target center has been changed due to joystick event, we can not use this to calibrate the target position.



Sample Code:

```
void moveTargetAngle()
{
	if( abs(joystickForwards) > something )
	{
		movementAnglePatch = (joystickForwards * something);
	}

	if( abs(joystickLeft) > something) )
	{
		turningDifferencial = (joystickLeft * something);
	}
}
```



Sample Code:

```
void correctUsingWheels()
{
	if( abs(sensorAngle - targetAngle) < something)
	{
		if(abs(sensorAngle - targetAngle) > something)
		{
			/* Note the sign of the turning differencial */
			applyToRightWheel(wheelSpeed() + turningDifferencial);
			applyToLeftWheel(wheelSpeed() - turningDifferencial);
		}
	}
}

float wheelSpeed()
{
	float targetAngle = calibratedStableAngle + movementAnglePatch;
	Return (sensorAngle – targetAngle) * something
}
```

Global Semaphores

Sensor Inputs

- Joystick Forwards
- Sensor Angle
- Joystick Left
- XAcceleration

Movement Angles and Adjustments

- Calibrated Stable Angle
- Turning differential
- Movement Angle Patch