Joint control

For a serial multi-joint robot, the joint control is to control the variables of each joint of the robot arm so as to make each joint reaches a target position at a certain speed.

1 Single joint control

1.1 Sending the angle of single joint

SendOneAngle(int jointNo, int angle, int speed)
Return value: none
Parameter description: Parameter 1: joint number (1-6), Parameter 2: angle (ranging from -170° to 170°), and Parameter 3: speed (0-100)
Case:

```
mc.SendOneAngle(1, 100, 70); ## 9.3.2 Multi-joint control
```

2 Multiple joint control

2.1 Get the angles of all joints

GetAngles()
Return value: return an array of int type, int[], length: 6
Description of parameters: none
Case:

```
var recv = mc.GetAngles();
```

2.2 send the angles of all joints

SendAngles(int[] angles, int speed)
Return value: none
Parameter description: Parameter 1: the angles of all joints (ranging from -170° to 170°), and Parameter 2: speed (0-100)
Case:

```
int[] angles = new[] {100, 100, 100, 100, 100, 100};
mc.SendAngles(angles ,30);
int[] angles = new[] {100, 100, 100, 100, 100, 100};
mc.SendAngles(angles,30);
```

3 Complete use cases

The program.cs in the project is a complete use case program, which can be modified as needed on this basis.

```
using System;
```

```
namespace Mycobot.csharp
{
    class Test
    {
        static void Main(string[] args)
        {
            MyCobot mc = new MyCobot("/dev/ttyUSB0");
            mc.Open();
            // int[] angles = new[] {100, 100, 100, 100, 100, 100};
            // mc.SendAngles(angles, 50);
            // Thread.Sleep(5000);
            // var recv = mc.GetAngles();
            // foreach (var v in recv)
            // {
                // Console.WriteLine(v);
            // }

            // int[] coords = new[] {160, 160, 160, 0, 0, 0};
            // mc.SendCoords(coords, 90, 1);
            // Thread.Sleep(5000);
            // var recv = mc.GetCoords();
            // foreach (var v in recv)
            // {
                // Console.WriteLine(v);
            // }

            mc.SendOneAngle(1, 100,70);

            // byte[] setColor = {0xfe, 0xfe, 0x05, 0x6a, 0xff, 0x00, 0x00,
0xfa};
            mc.Close();
        }
    }
}
```