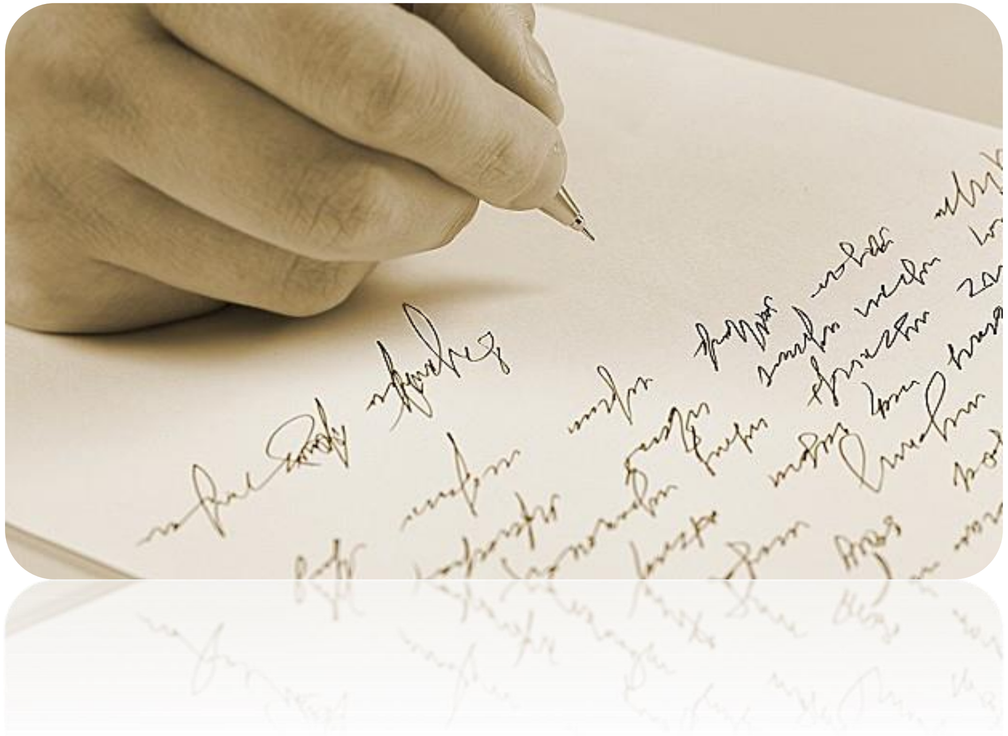
CATCH ME IF YOU CAN… Part 2

Advanced Mechatronics :

Propeller Mini Project



Presented By:

Federico Gregori

Karim Chamaa

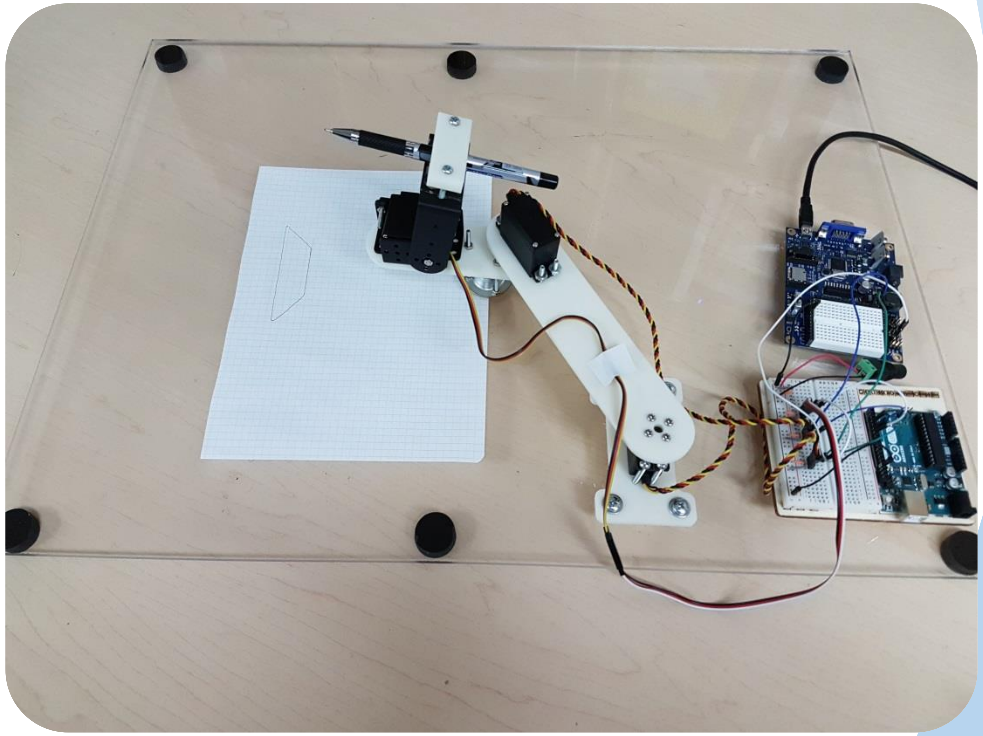
Presented to:
Dr. Vikram Kapila

Outline

- Introduction
- Improvements
- Circuit Design
- Coding
- System Speed
- Comparisons and Results
- Future Improvements
- Conclusion

Introduction

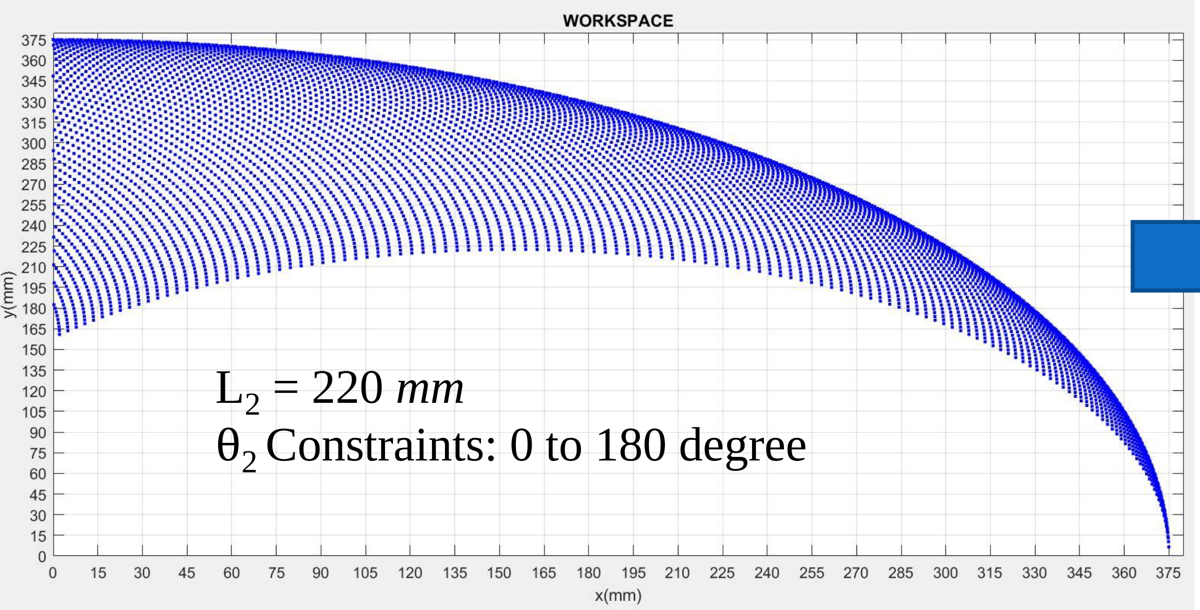
- Trying to achieve better results with design improvements
- Collecting data by LabVIEW and transferring them to the microcontrollers
- Comparing results obtained using Arduino or Propeller microcontroller



Improvements

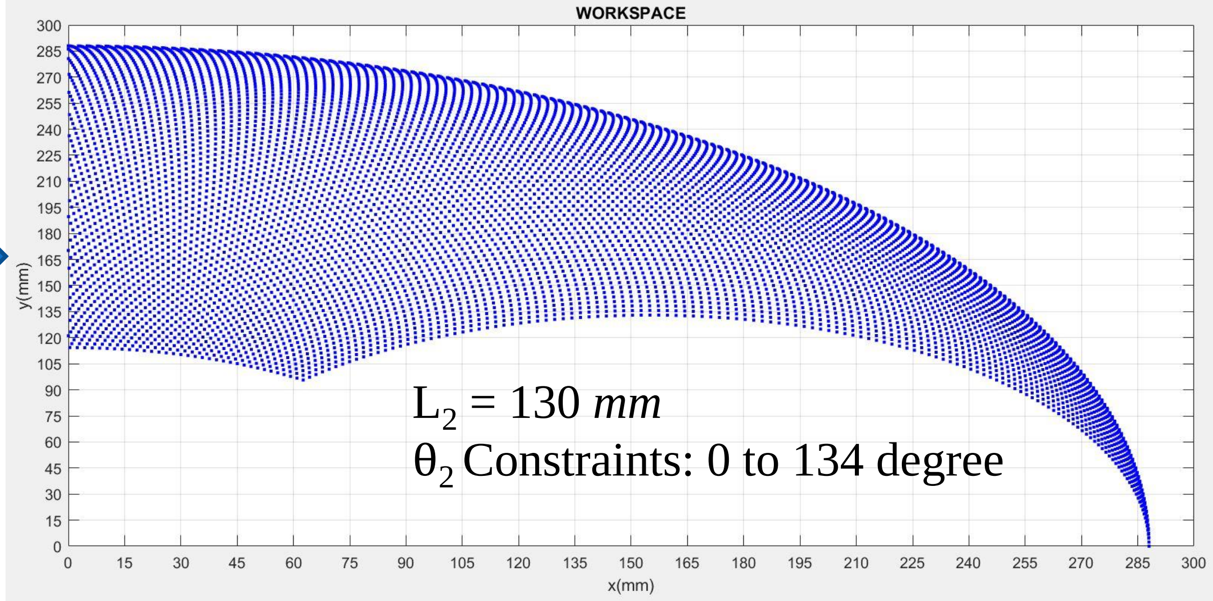
- **Achieved** a stable system decreasing the length of the second link
- **Obtained** a larger workspace area
- **Increased** the torque of the servos by supplying a 10A power source
- **Improved** the simultaneity of the commands using two cogs in parallel
- **Decreased** the friction using a ballpoint pen with a smaller diameter

Improvements



$L_2 = 220\ mm$

θ_2 Constraints: 0 to 180 degree

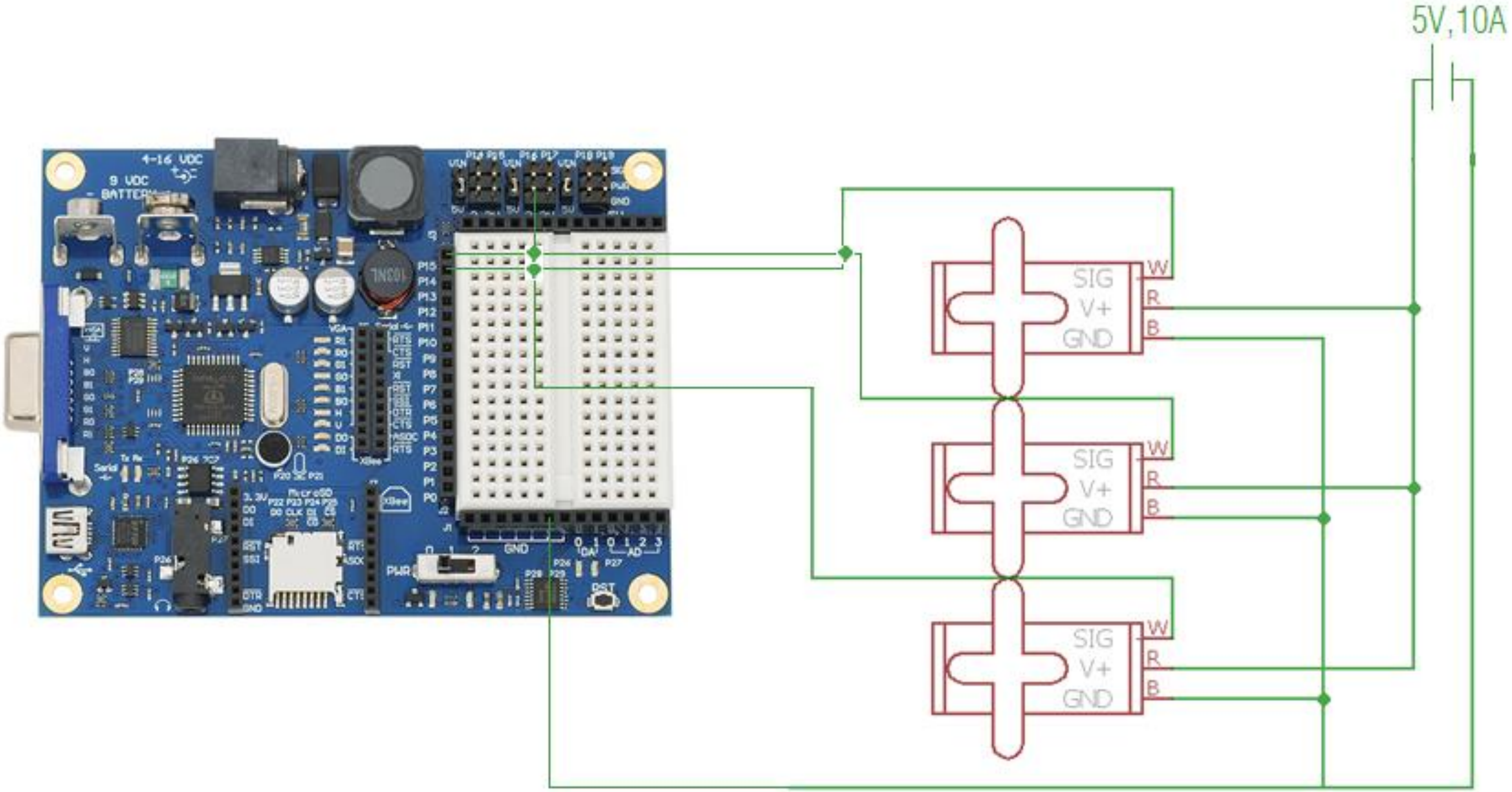


$L_2 = 130\ mm$

θ_2 Constraints: 0 to 134 degree

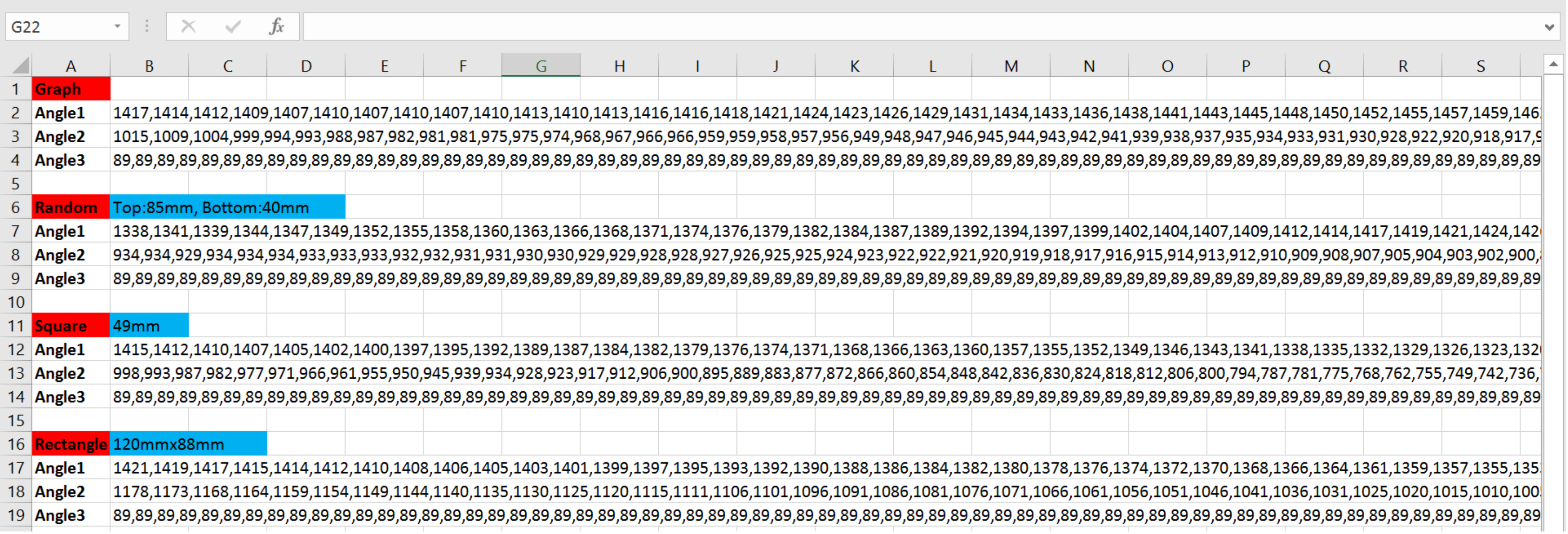
- Graphs obtained through a Matlab simulation

Circuit Design



Coding

STEP1 → Acquiring data **Automatically** from LabVIEW



G22

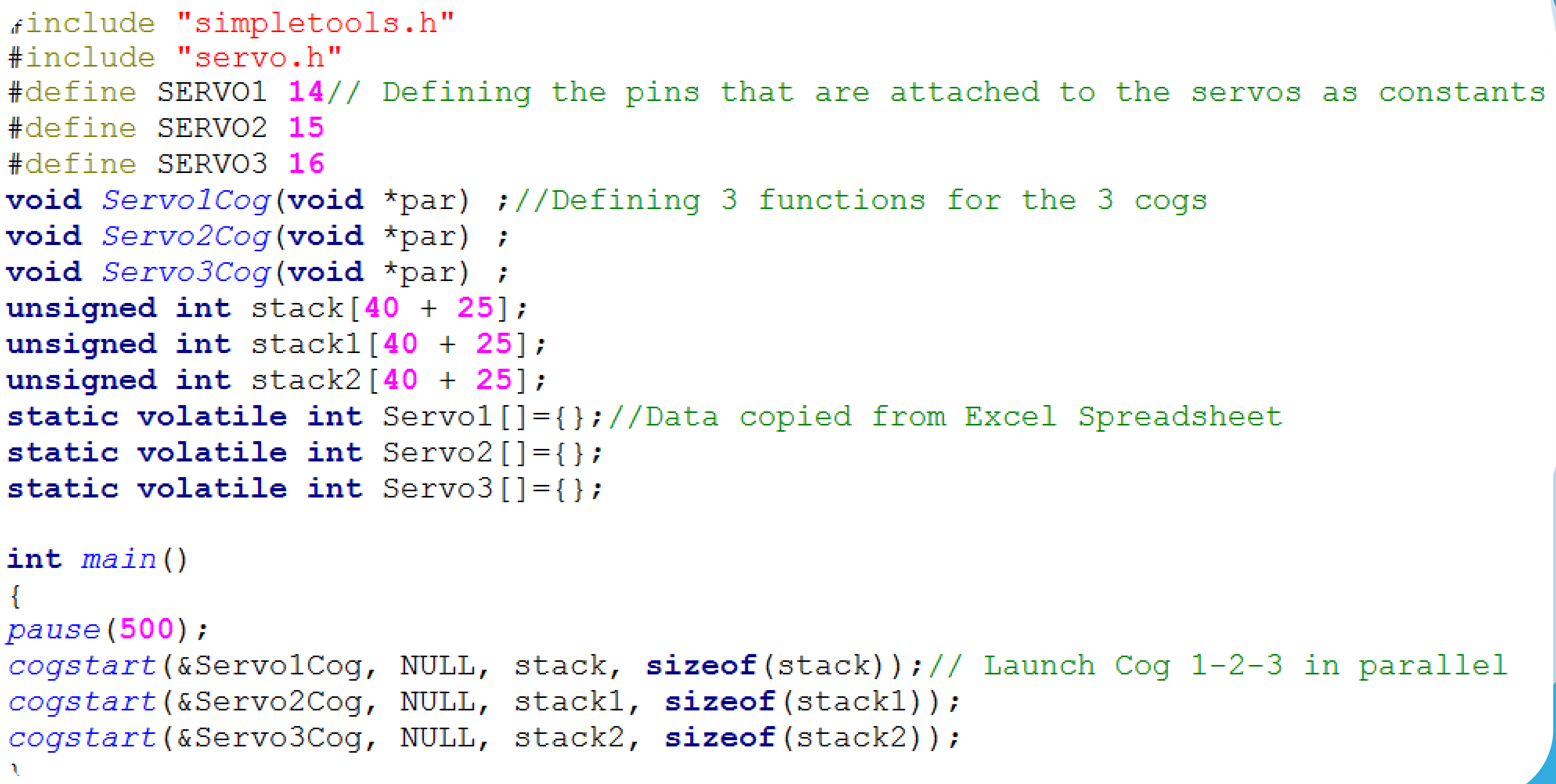
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	**Graph**																		
2	**Angle1**	1417,1414,1412,1409,1407,1410,1407,1410,1407,1410,1413,1410,1413,1416,1416,1418,1421,1424,1423,1426,1429,1431,1434,1433,1436,1438,1441,1443,1445,1448,1450,1452,1455,1457,1459,146																	
3	**Angle2**	1015,1009,1004,999,994,993,988,987,982,981,981,975,975,974,968,967,966,966,959,959,958,957,956,949,948,947,946,945,944,943,942,941,939,938,937,935,934,933,931,930,928,922,920,918,917,9																	
4	**Angle3**	89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89																	
5																			
6	**Random**	Top:85mm, Bottom:40mm																	
7	**Angle1**	1338,1341,1339,1344,1347,1349,1352,1355,1358,1360,1363,1366,1368,1371,1374,1376,1379,1382,1384,1387,1389,1392,1394,1397,1399,1402,1404,1407,1409,1412,1414,1417,1419,1421,1424,142																	
8	**Angle2**	934,934,929,934,934,934,933,933,933,932,932,931,931,930,930,929,929,928,928,927,926,925,925,924,923,922,922,921,920,919,918,917,916,915,914,913,912,910,909,908,907,905,904,903,902,900,																	
9	**Angle3**	89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89																	
10																			
11	**Square**	49mm																	
12	**Angle1**	1415,1412,1410,1407,1405,1402,1400,1397,1395,1392,1389,1387,1384,1382,1379,1376,1374,1371,1368,1366,1363,1360,1357,1355,1352,1349,1346,1343,1341,1338,1335,1332,1329,1326,1323,132																	
13	**Angle2**	998,993,987,982,977,971,966,961,955,950,945,939,934,928,923,917,912,906,900,895,889,883,877,872,866,860,854,848,842,836,830,824,818,812,806,800,794,787,781,775,768,762,755,749,742,736,																	
14	**Angle3**	89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89																	
15																			
16	**Rectangle**	120mmx88mm																	
17	**Angle1**	1421,1419,1417,1415,1414,1412,1410,1408,1406,1405,1403,1401,1399,1397,1395,1393,1392,1390,1388,1386,1384,1382,1380,1378,1376,1374,1372,1370,1368,1366,1364,1361,1359,1357,1355,135																	
18	**Angle2**	1178,1173,1168,1164,1159,1154,1149,1144,1140,1135,1130,1125,1120,1115,1111,1106,1101,1096,1091,1086,1081,1076,1071,1066,1061,1056,1051,1046,1041,1036,1031,1025,1020,1015,1010,100																	
19	**Angle3**	89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89																	

- Data:
  - Formatted(10th of a degree)
  - Separated by commas
  - Transposed

→ Data should be copied and pasted in SimpleIDE

Coding Propeller servo_angle (Multi Cogs)

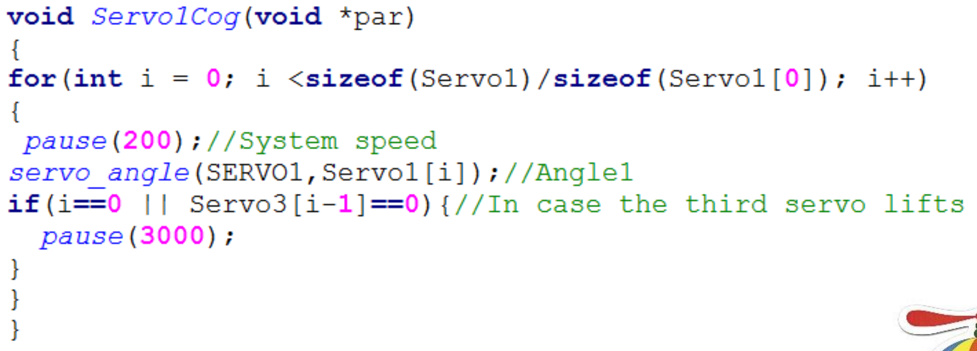
STEP2 → Propeller Code-Part1



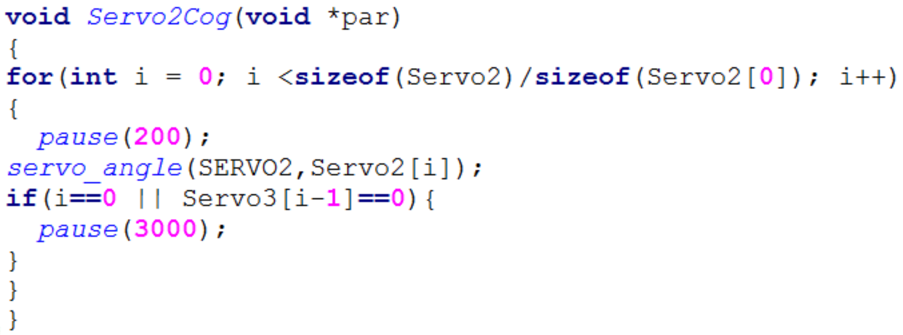
```
#include "simpletools.h"
#include "servo.h"
#define SERVO1 14// Defining the pins that are attached to the servos as constants
#define SERVO2 15
#define SERVO3 16
void Servo1Cog(void *par) ;//Defining 3 functions for the 3 cogs
void Servo2Cog(void *par) ;
void Servo3Cog(void *par) ;
unsigned int stack[40 + 25];
unsigned int stack1[40 + 25];
unsigned int stack2[40 + 25];
static volatile int Servo1[]={};//Data copied from Excel Spreadsheet
static volatile int Servo2[]={};
static volatile int Servo3[]={};

int main()
{
pause(500);
cogstart(&Servo1Cog, NULL, stack, sizeof(stack));// Launch Cog 1-2-3 in parallel
cogstart(&Servo2Cog, NULL, stack1, sizeof(stack1));
cogstart(&Servo3Cog, NULL, stack2, sizeof(stack2));
}
```

Coding Propeller servo_angle (Multi Cogs)

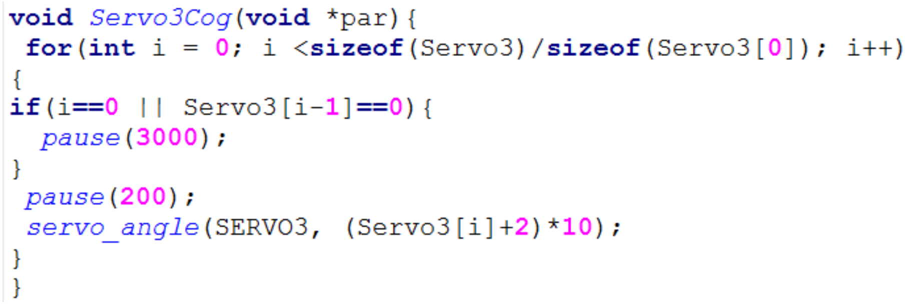
STEP2 → Propeller Code-Part2



```
void Servo1Cog(void *par)
{
for(int i = 0; i <sizeof(Servo1)/sizeof(Servo1[0]); i++)
{
 pause(200);//System speed
servo_angle(SERVO1,Servo1[i]);//Angle1
if(i==0 || Servo3[i-1]==0){//In case the third servo lifts
  pause(3000);
}
}
}
```



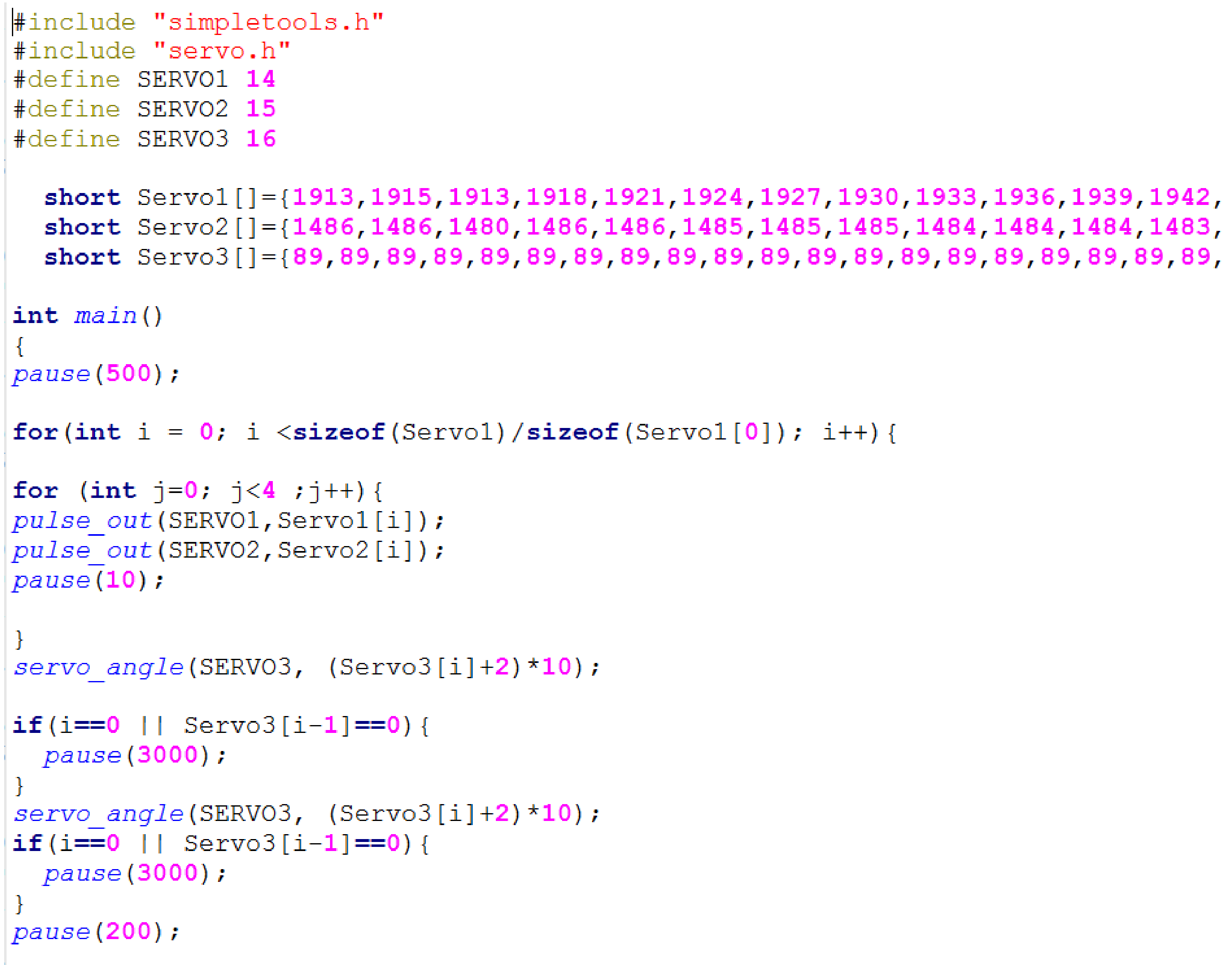
```
void Servo2Cog(void *par)
{
for(int i = 0; i <sizeof(Servo2)/sizeof(Servo2[0]); i++)
{
  pause(200);
servo_angle(SERVO2,Servo2[i]);
if(i==0 || Servo3[i-1]==0){
  pause(3000);
}
}
}
```



```
void Servo3Cog(void *par){
 for(int i = 0; i <sizeof(Servo3)/sizeof(Servo3[0]); i++)
{
if(i==0 || Servo3[i-1]==0){
  pause(3000);
}
 pause(200);
 servo_angle(SERVO3, (Servo3[i]+2)*10);
}
}
```

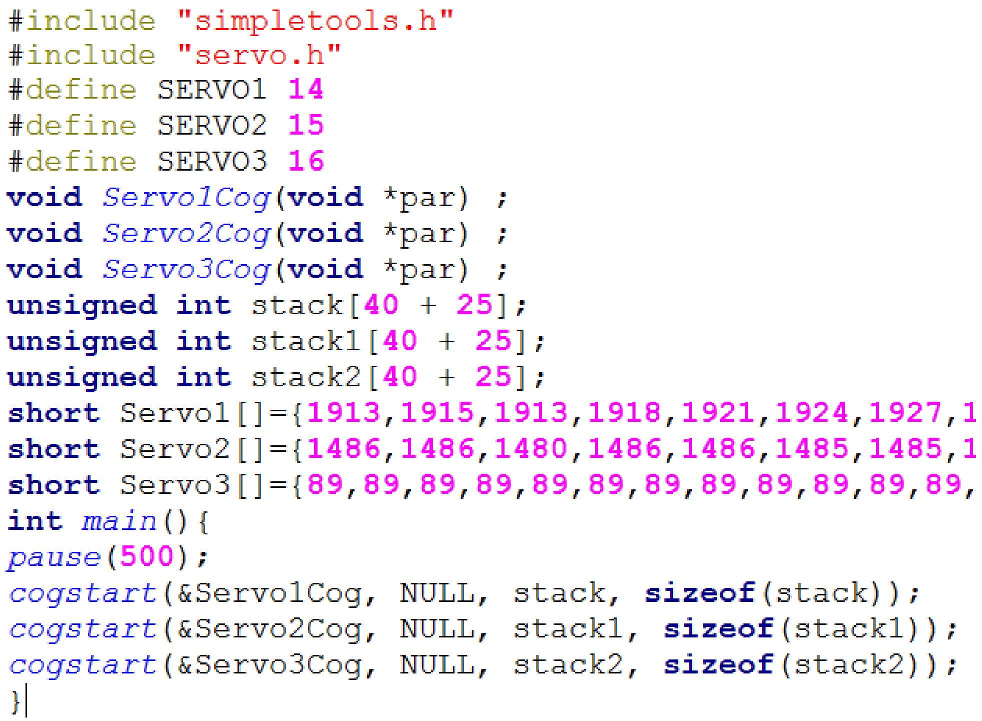
- Each COG is controlling the position of a servo motor
- Total pause time in each COG is maintained equal

Coding Propeller pulse_out (Single Cog)

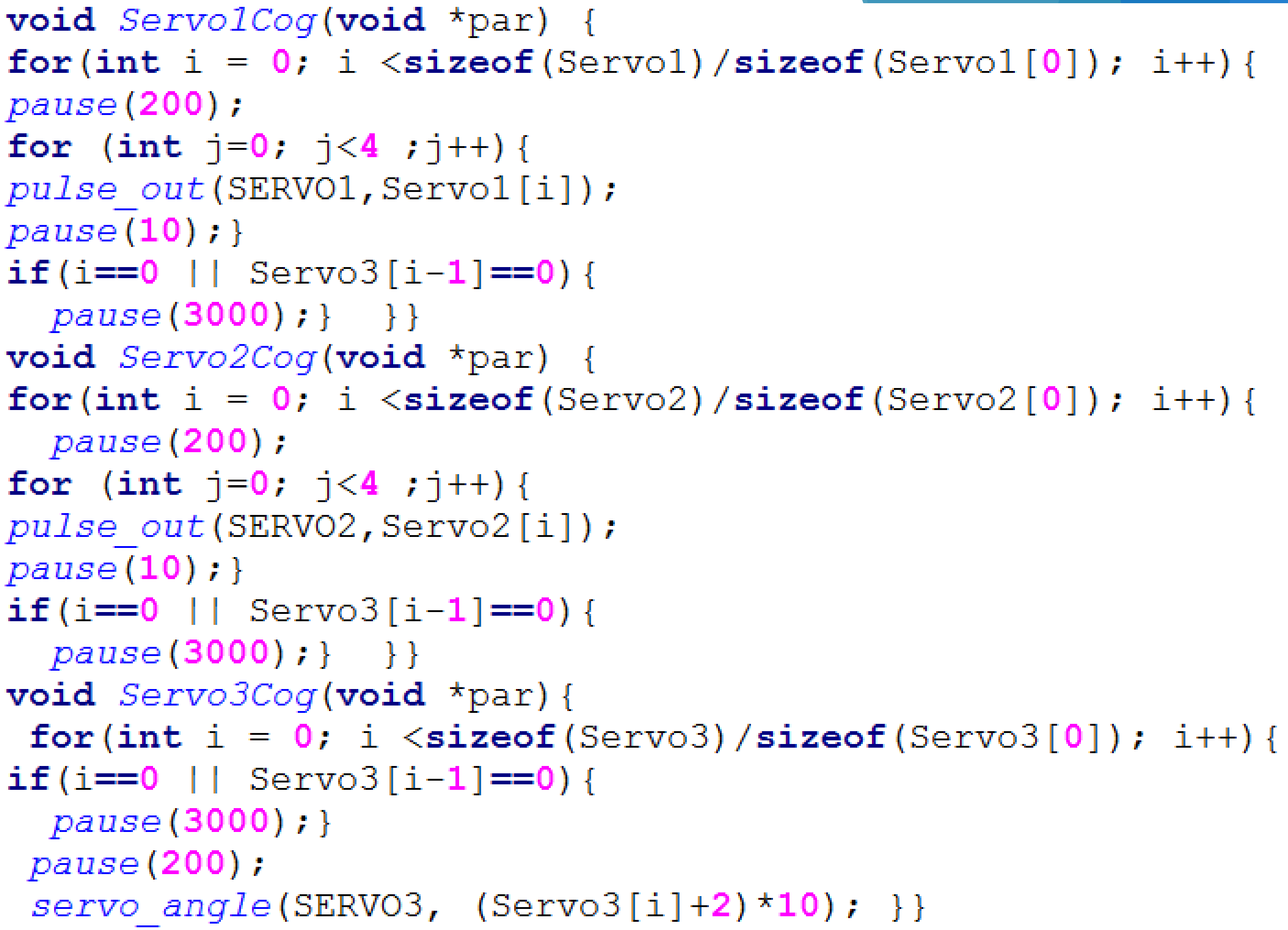


```
#include "simpletools.h"
#include "servo.h"
#define SERVO1 14
#define SERVO2 15
#define SERVO3 16

  short Servo1[]={1913,1915,1913,1918,1921,1924,1927,1930,1933,1936,1939,1942,
  short Servo2[]={1486,1486,1480,1486,1486,1485,1485,1485,1484,1484,1484,1483,
  short Servo3[]={89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,

int main()
{
pause(500);

for(int i = 0; i <sizeof(Servo1)/sizeof(Servo1[0]); i++){

for (int j=0; j<4 ;j++){
pulse_out(SERVO1,Servo1[i]);
pulse_out(SERVO2,Servo2[i]);
pause(10);

}
servo_angle(SERVO3, (Servo3[i]+2)*10);

if(i==0 || Servo3[i-1]==0){
  pause(3000);
}
servo_angle(SERVO3, (Servo3[i]+2)*10);
if(i==0 || Servo3[i-1]==0){
  pause(3000);
}
pause(200);
```

Coding Propeller pulse_out (Multi Cogs)

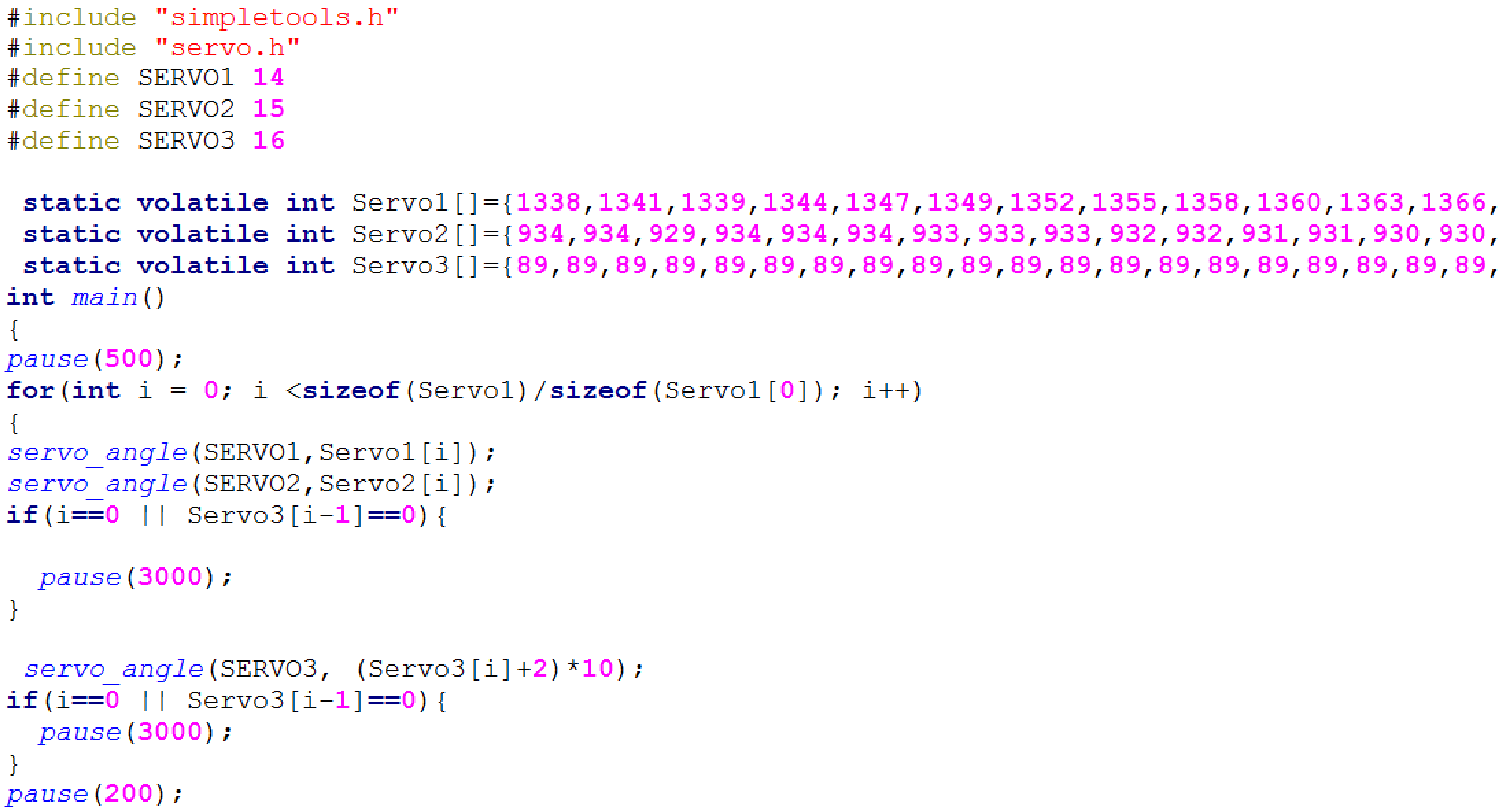


```
#include "simpletools.h"
#include "servo.h"
#define SERVO1 14
#define SERVO2 15
#define SERVO3 16
void Servo1Cog(void *par) ;
void Servo2Cog(void *par) ;
void Servo3Cog(void *par) ;
unsigned int stack[40 + 25];
unsigned int stack1[40 + 25];
unsigned int stack2[40 + 25];
short Servo1[]={1913,1915,1913,1918,1921,1924,1927,1
short Servo2[]={1486,1486,1480,1486,1486,1485,1485,1
short Servo3[]={89,89,89,89,89,89,89,89,89,89,89,89,
int main(){
pause(500);
cogstart(&Servo1Cog, NULL, stack, sizeof(stack));
cogstart(&Servo2Cog, NULL, stack1, sizeof(stack1));
cogstart(&Servo3Cog, NULL, stack2, sizeof(stack2));
}
```



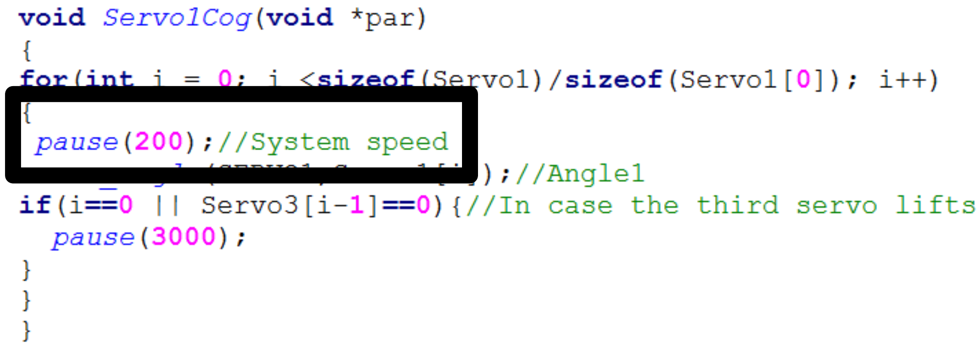
```
void Servo1Cog(void *par) {
for(int i = 0; i <sizeof(Servo1)/sizeof(Servo1[0]); i++){
pause(200);
for (int j=0; j<4 ;j++){
pulse_out(SERVO1,Servo1[i]);
pause(10);}
if(i==0 || Servo3[i-1]==0){
  pause(3000);}  }}
void Servo2Cog(void *par) {
for(int i = 0; i <sizeof(Servo2)/sizeof(Servo2[0]); i++){
  pause(200);
for (int j=0; j<4 ;j++){
pulse_out(SERVO2,Servo2[i]);
pause(10);}
if(i==0 || Servo3[i-1]==0){
  pause(3000);}  }}
void Servo3Cog(void *par){
 for(int i = 0; i <sizeof(Servo3)/sizeof(Servo3[0]); i++){
if(i==0 || Servo3[i-1]==0){
  pause(3000);}
 pause(200);
 servo_angle(SERVO3, (Servo3[i]+2)*10);  }}
```

Coding Propeller servo_angle (Single Cog)

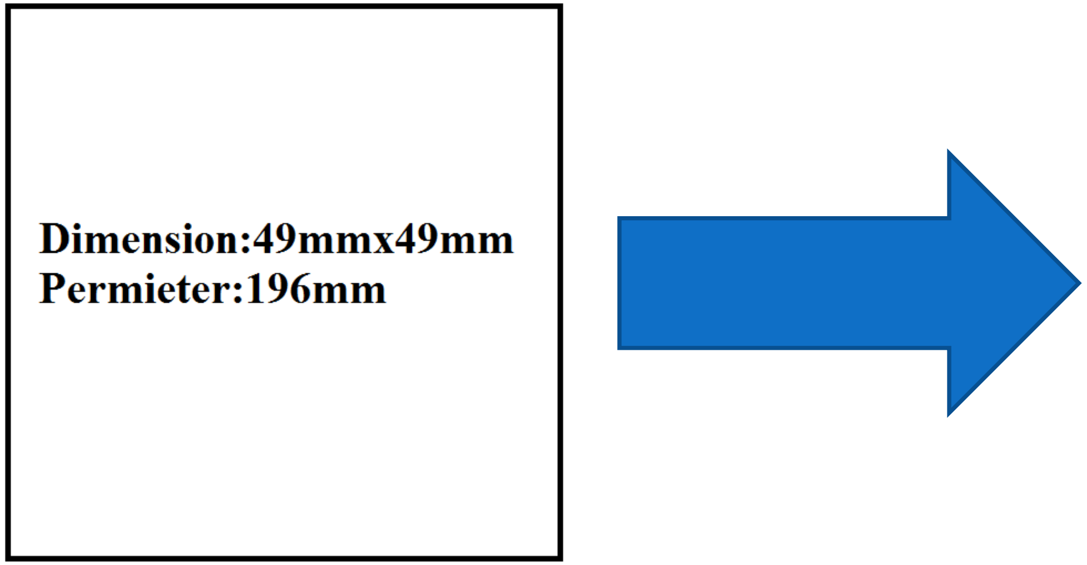


```
#include "simpletools.h"
#include "servo.h"
#define SERVO1 14
#define SERVO2 15
#define SERVO3 16

 static volatile int Servo1[]={1338,1341,1339,1344,1347,1349,1352,1355,1358,1360,1363,1366,
 static volatile int Servo2[]={934,934,929,934,934,934,933,933,933,932,932,931,931,930,930,
 static volatile int Servo3[]={89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,89,
int main()
{
pause(500);
for(int i = 0; i <sizeof(Servo1)/sizeof(Servo1[0]); i++)
{
servo_angle(SERVO1,Servo1[i]);
servo_angle(SERVO2,Servo2[i]);
if(i==0 || Servo3[i-1]==0){

  pause(3000);
}

 servo_angle(SERVO3, (Servo3[i]+2)*10);
if(i==0 || Servo3[i-1]==0){
  pause(3000);
}
pause(200);
```

System Speed



```
void Servo1Cog(void *par)
{
for(int i = 0; i <sizeof(Servo1)/sizeof(Servo1[0]); i++)
{
 pause(200);//System speed
 servo_angle(SERVO1,Servo1[i]);//Angle1
if(i==0 || Servo3[i-1]==0){//In case the third servo lifts
  pause(3000);
}
}
}
```

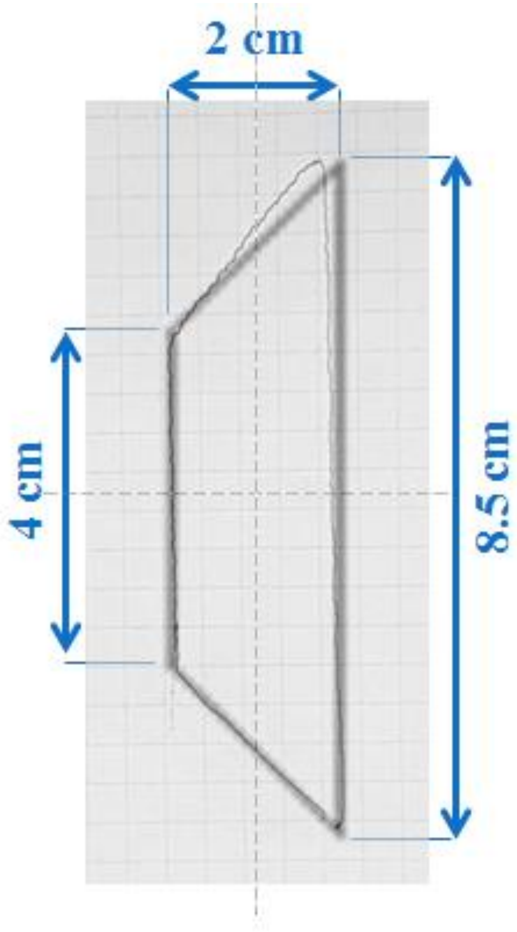


Dimension:49mmx49mm
Permieter:196mm

39.2 Seconds

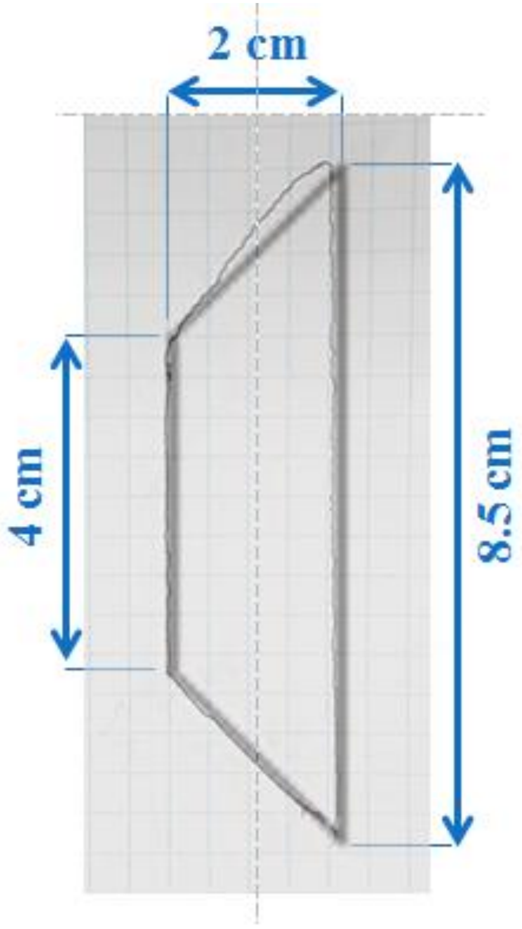
Comparison

LabVIEW + Arduino



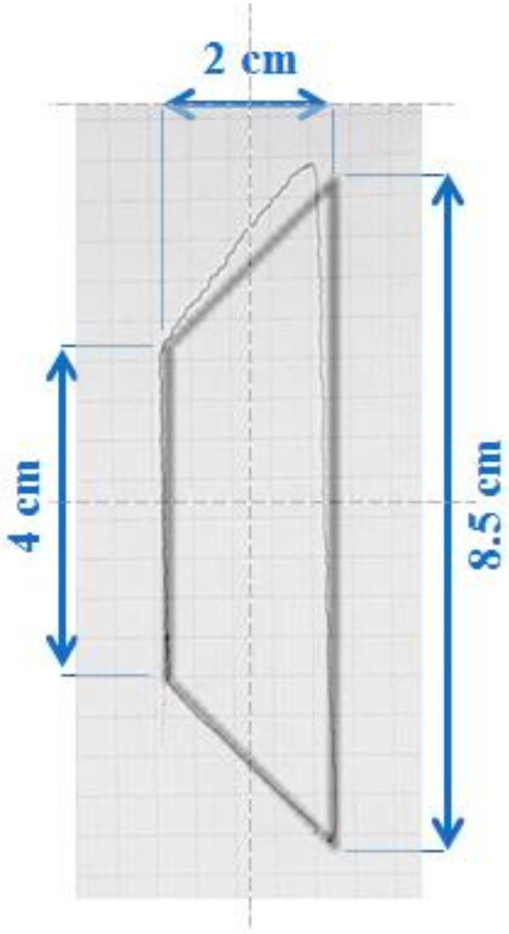
VS

Arduino

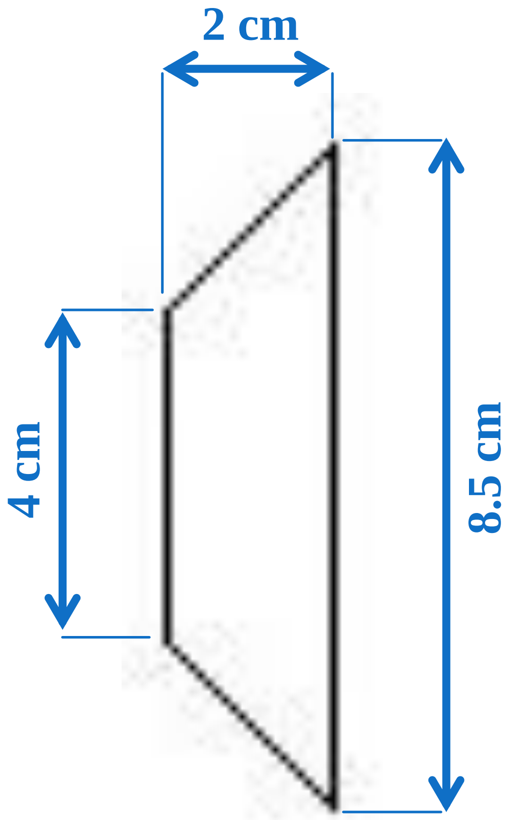


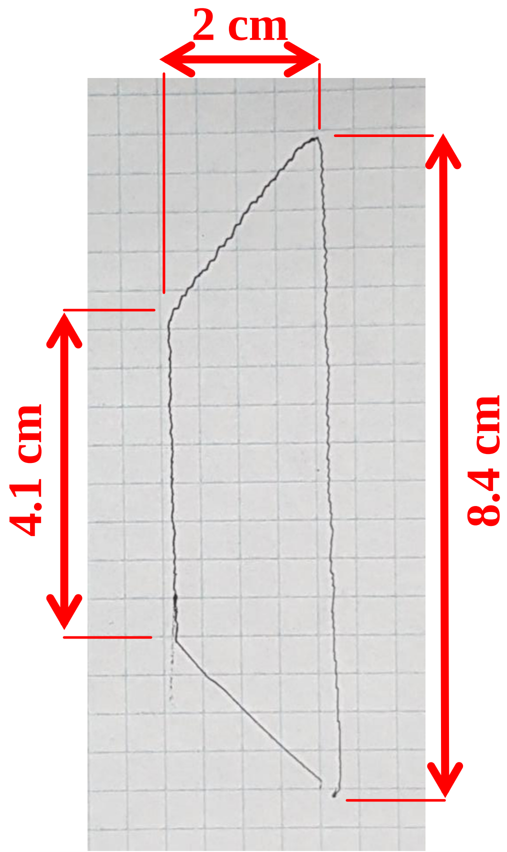
VS

Propeller



Results LabVIEW + Arduino



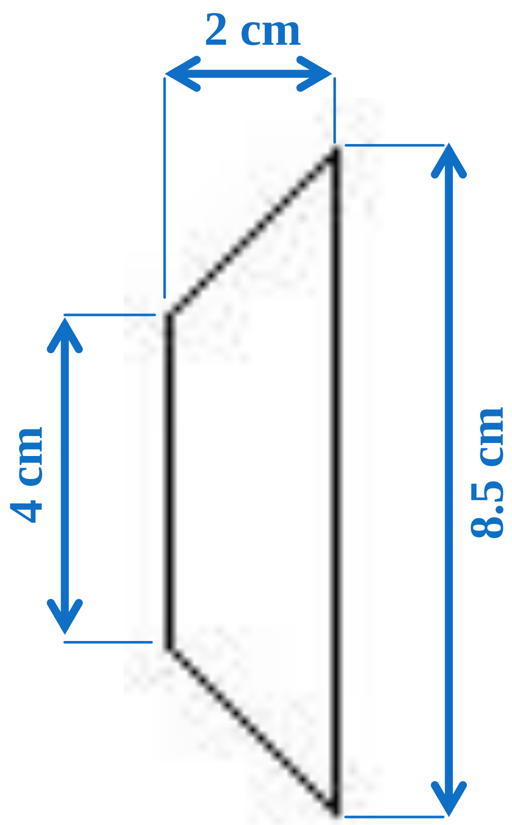


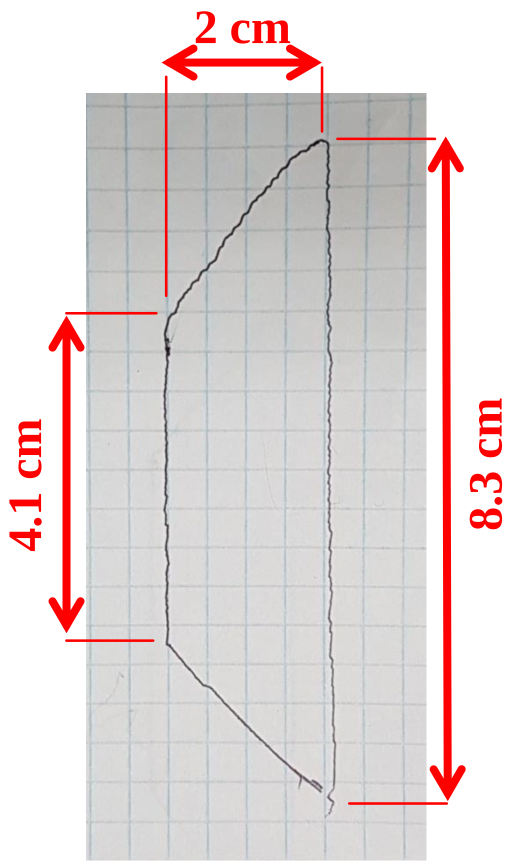
- The error evaluated is 1.2%*

$$\varepsilon = \frac{1}{3}\left(\frac{|l_1 - l_1^*|}{l_1^*} + \frac{|l_2 - l_2^*|}{l_2^*} + \frac{|l_3 - l_3^*|}{l_3^*}\right)$$

* The technique used is the mean value of the relative error of the three measurements.

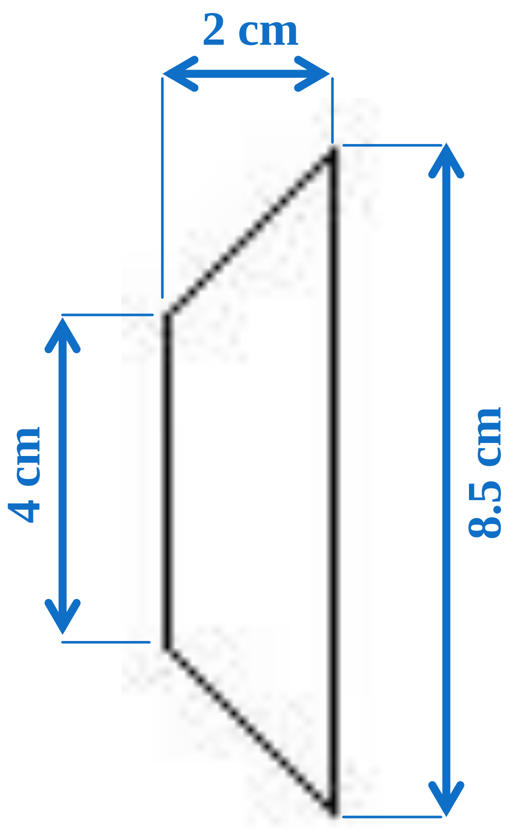
Arduino

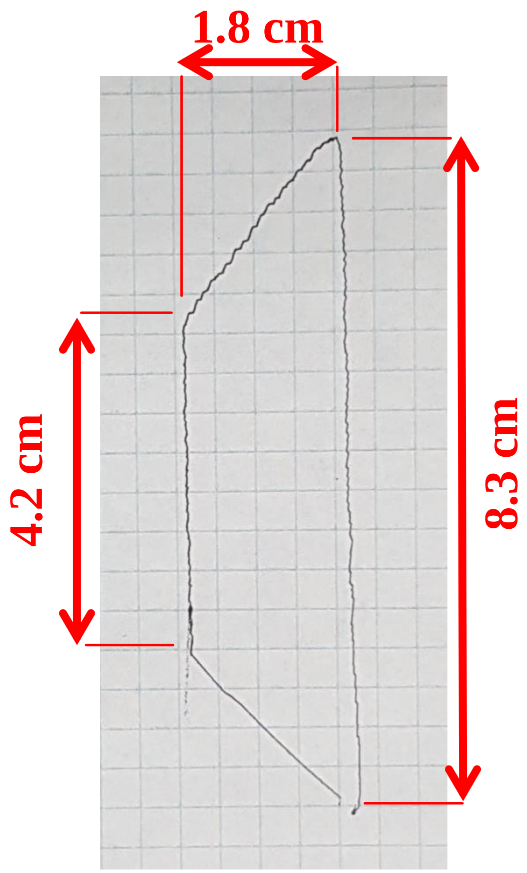




- The error evaluated is 1.6%
- Labview does not influence the output obtained with Arduino

Propeller Servo_angle (Multi Cogs)

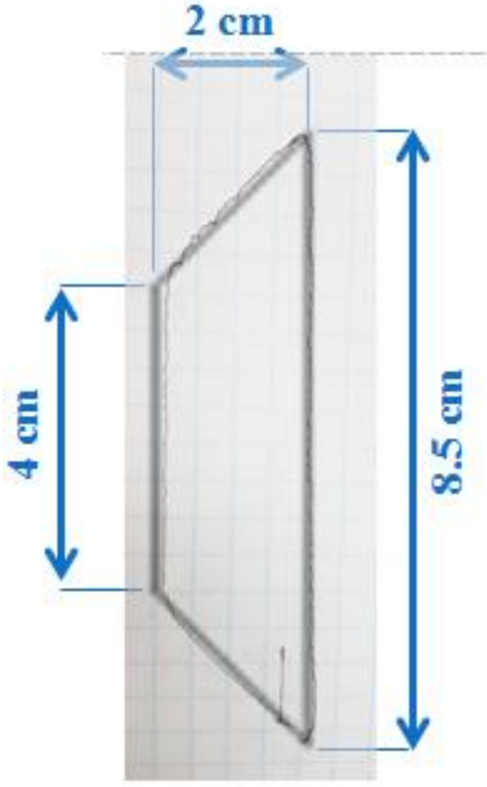




- The error evaluated is 5.8%
- Propeller provides a larger error than Arduino

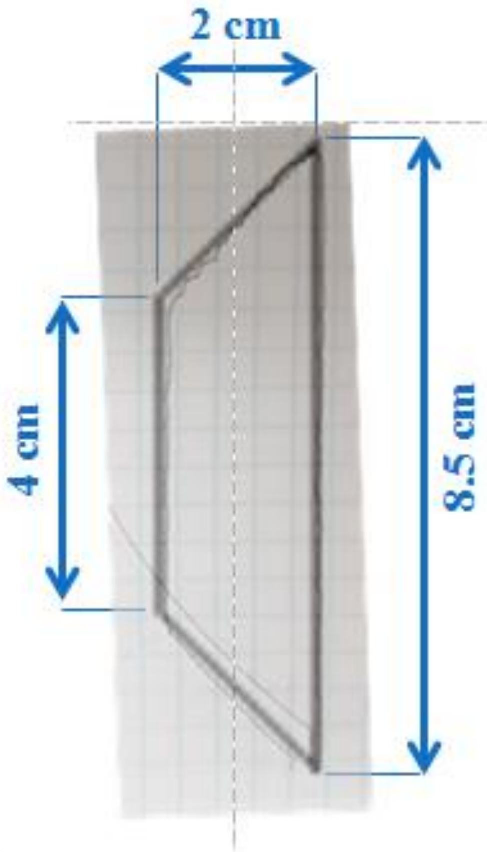
Propeller Comparison

pulse_out Single Cog



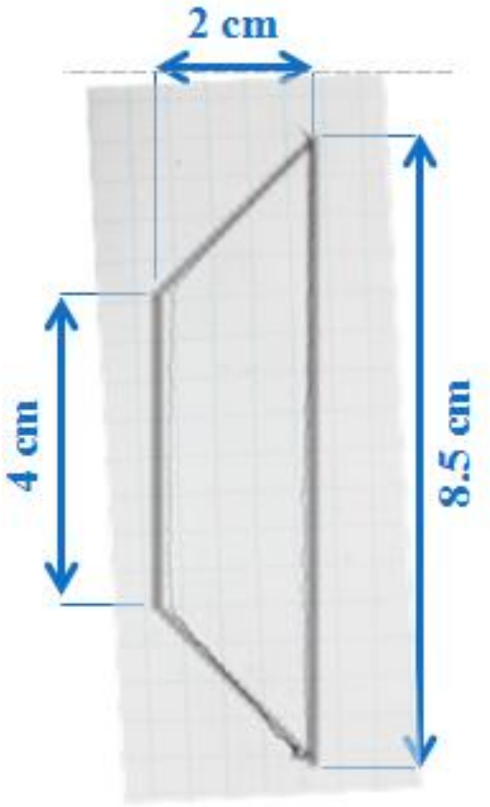
VS

pulse_out Multi Cogs

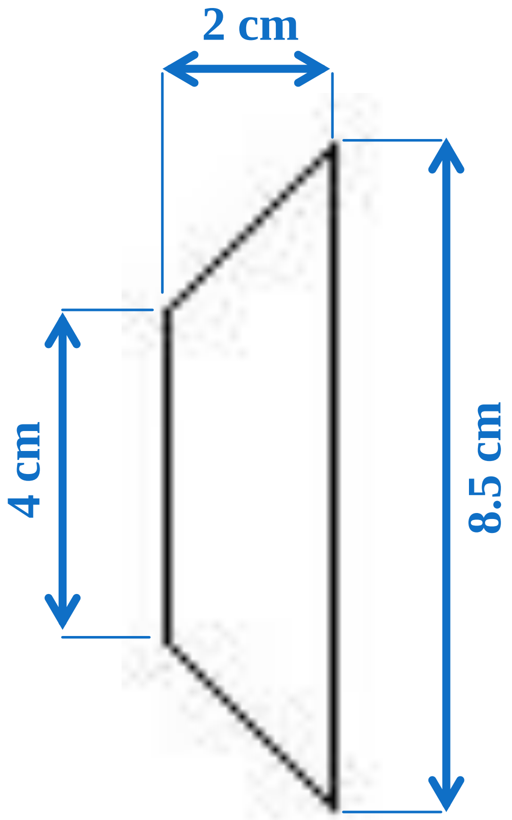


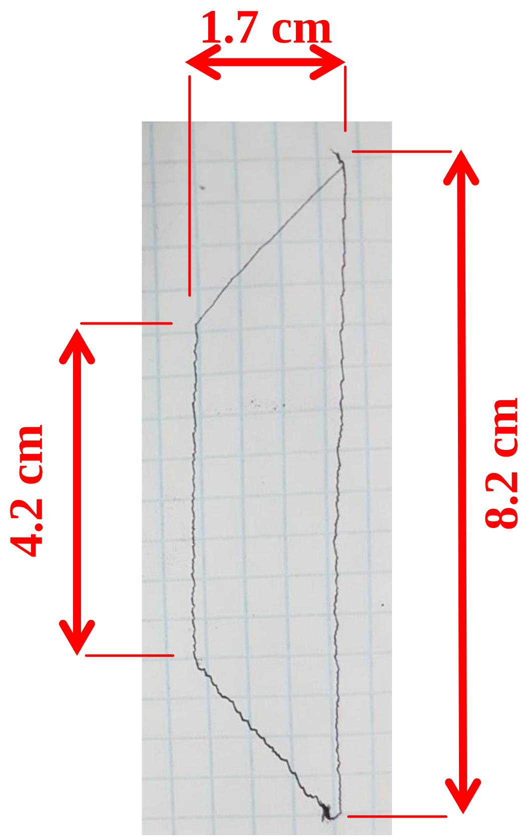
VS

servo_angle Single Cog



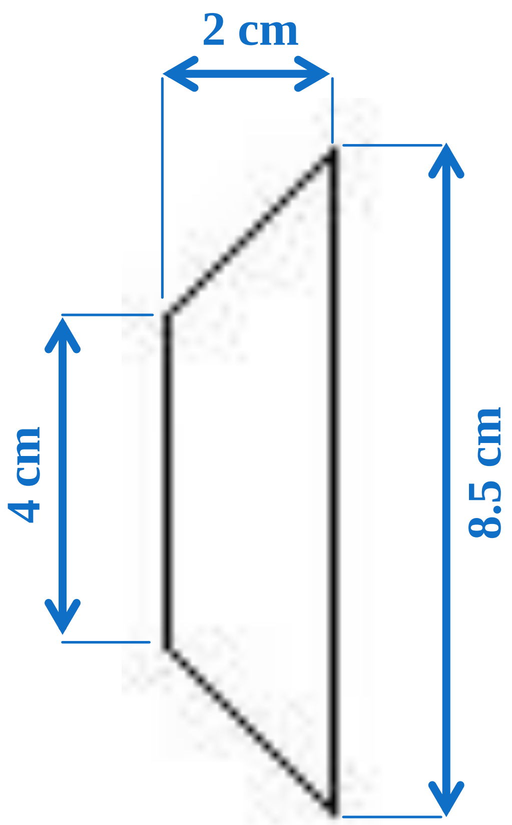
Servo_angle Single Cog

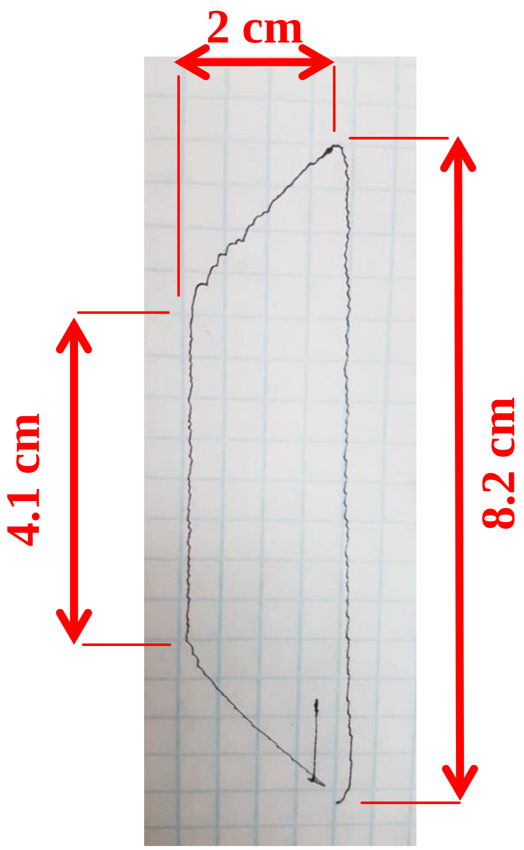




- The error evaluated is 7.8%
- Using a single core will decrease the accuracy of the system

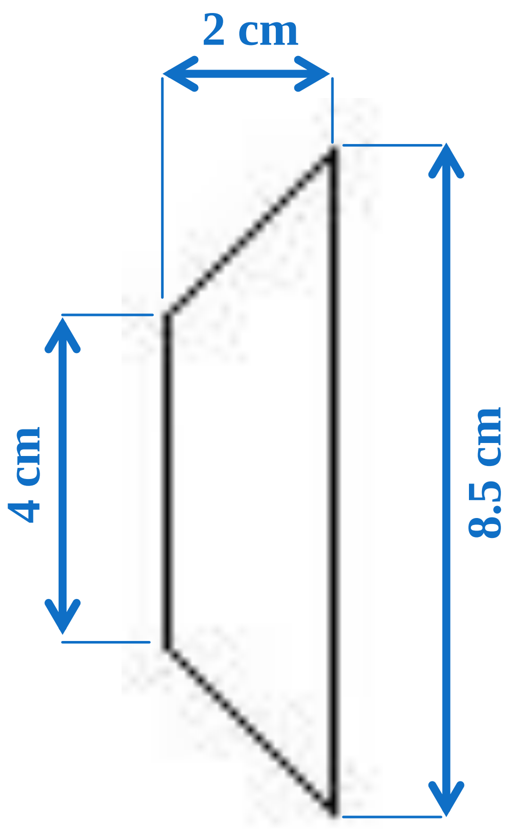
Pulse_out Single Cog

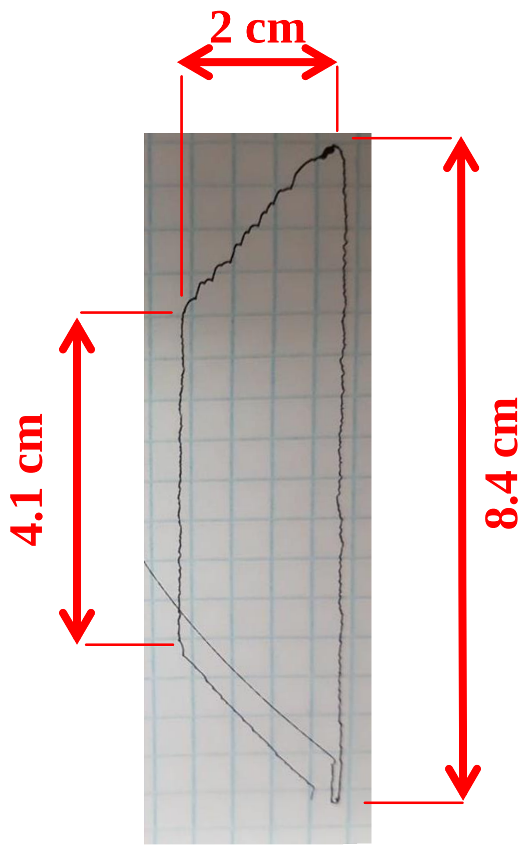




- The error evaluated is 2%
- Results comparable with those of Arduino
- Needed many calibration due to pulse out command
- Better shapes with faster loop but less accuracy

Pulse_out Multi Cogs





- The error evaluated is 1.2%
- Results obtained using multiple cogs are better than single cog but not really parallel
- Needed many calibration due to pulse out command
- Better shapes with faster loop but less accuracy

Conclusions

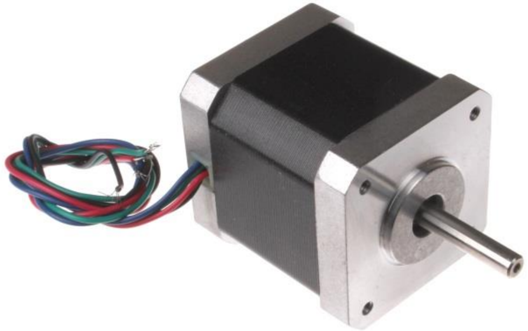
- We achieved better results by replacing the servo-angle command in a propeller code with a pulse-out and this is due to several reasons including:
  - Range of Angle in pulse out(2400-550=1850) bigger than the range of Angle in servo-angle(1800-0=1800) by 50 Angles.
  - Angles used in servo-angle function were rounded to the nearest angle.
- Coding using writeMicroseconds() in Arduino and pulse-out command in Propeller gave similar result. Slight percentage error between them is due to :
  - Error while measuring the percentage error
  - Precise scaling of the for loop.

Conclusions

- Servo motors do not offer a valid solution for the aim of the project
- More stable and accurate actuators are needed
- The results obtained by Arduino or Propeller are comparable
- An high current is necessary to run the servos properly
- Controlling Arduino directly from LabVIEW does not implicate worst results

Future Improvements

- Improving the accuracy and the stability of the system with stepper motors.



- Acquiring an image through the raspberry pi cam and processing it which will eliminate the need of LabVIEW.

Thank You

Questions ?