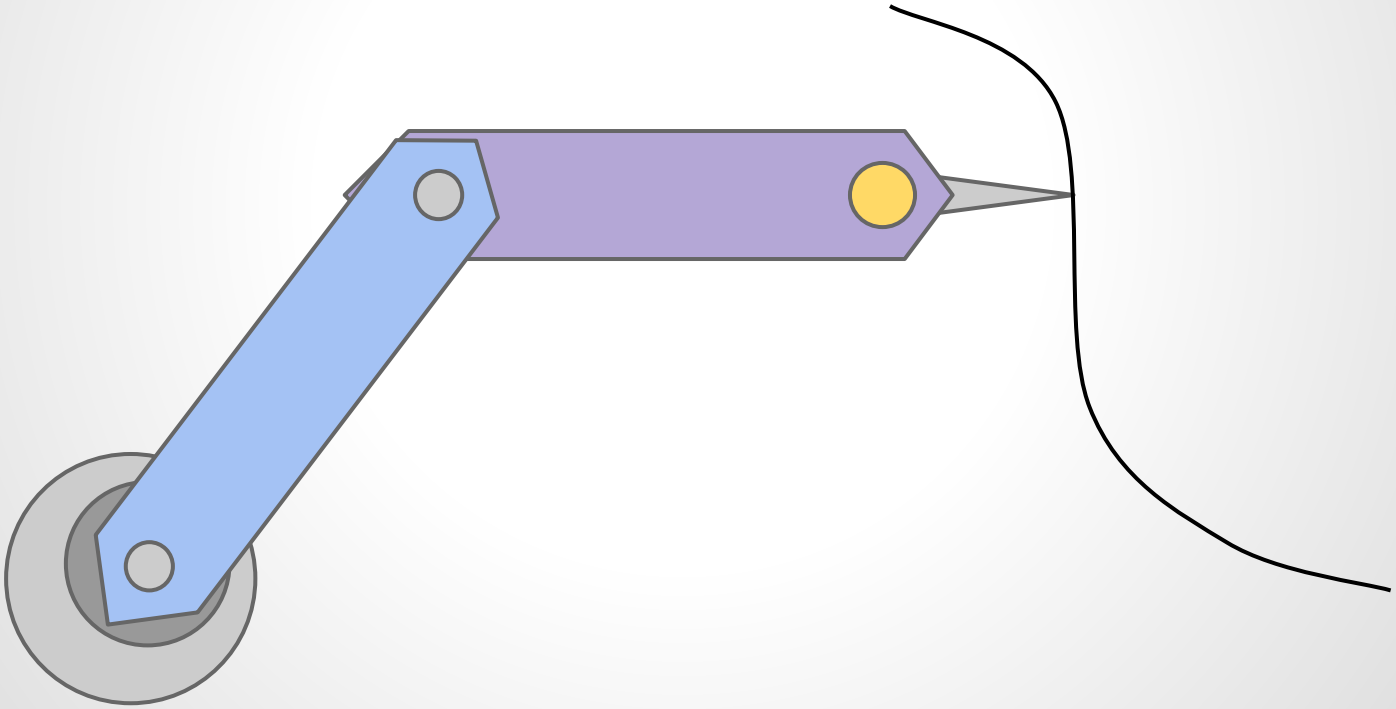
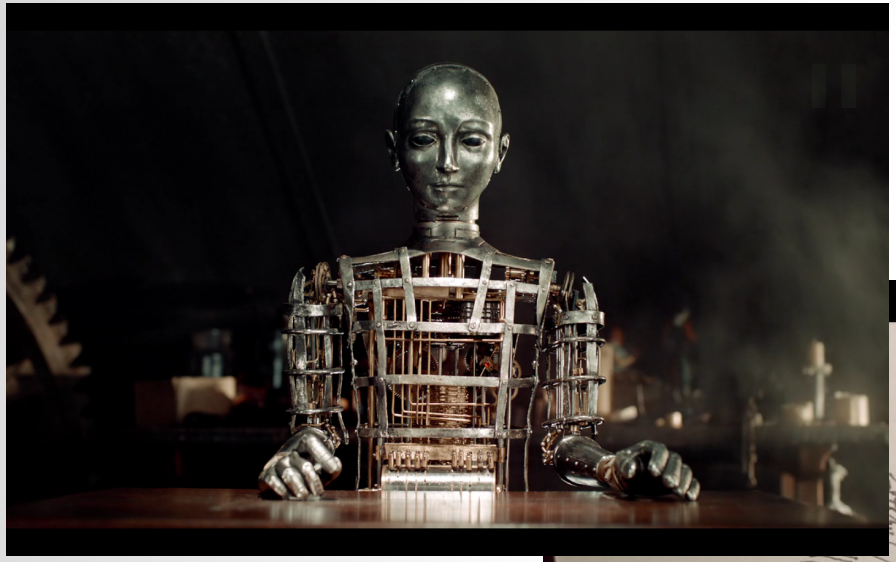
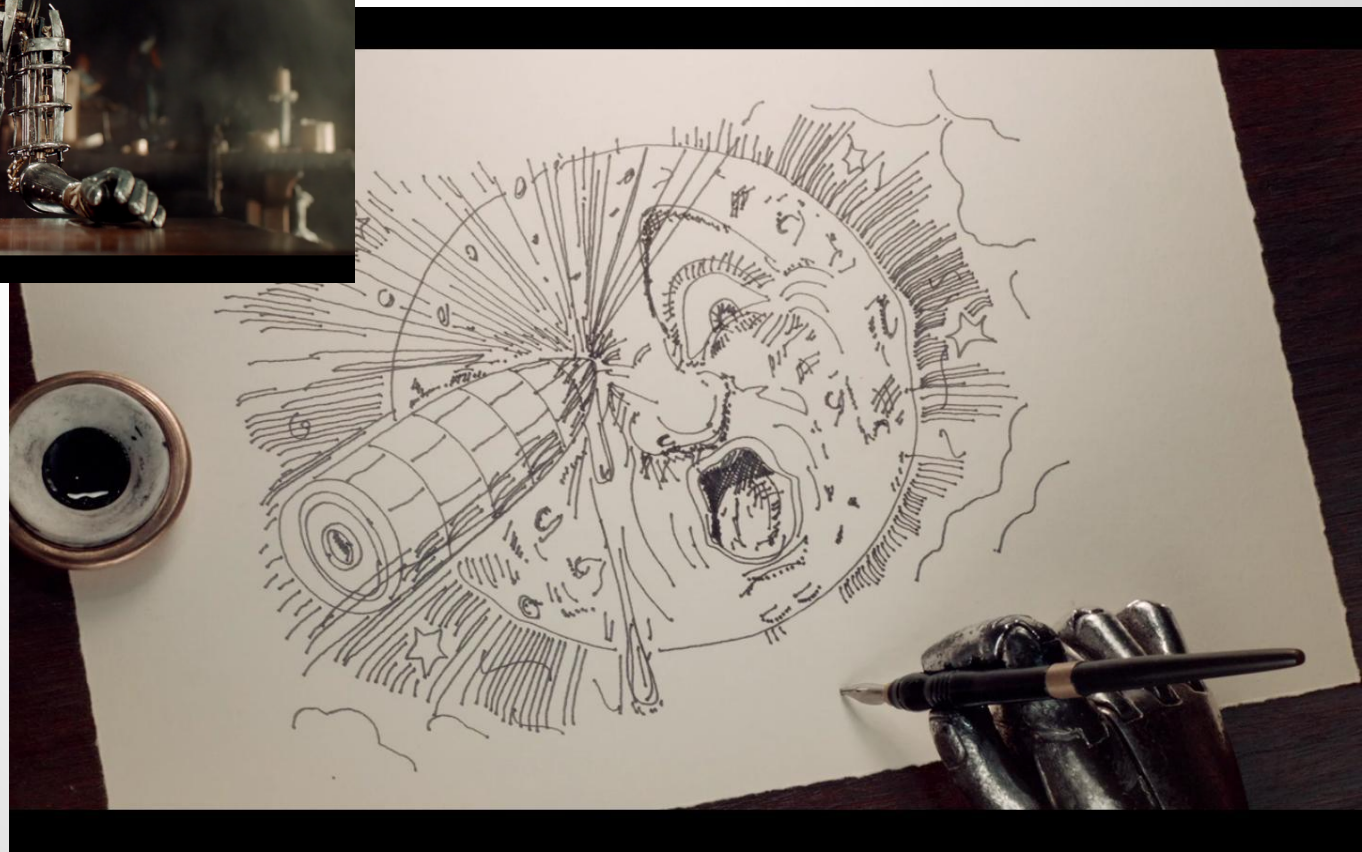
Robotic plotter arm



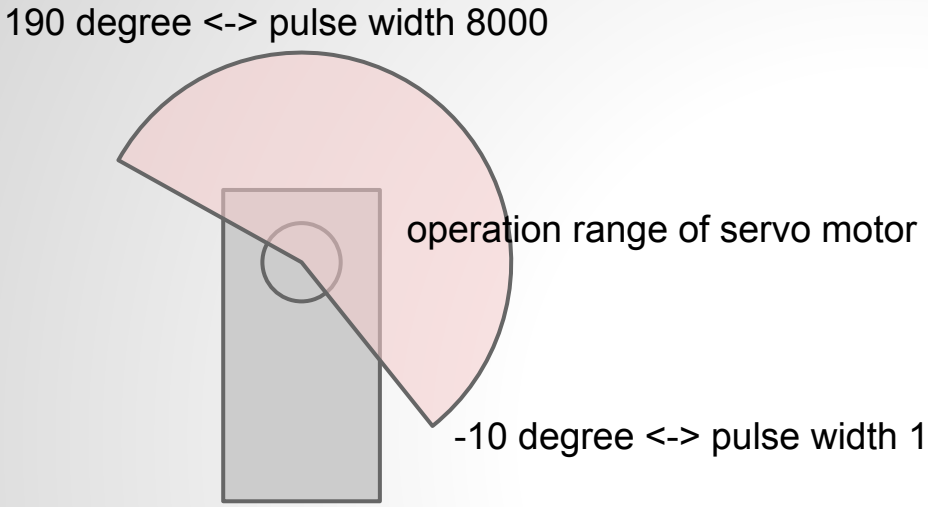
Human-like drawing arm is not an invention, it's a fiction



(novel) The Invention of Hugo Cabret
(film) Hugo

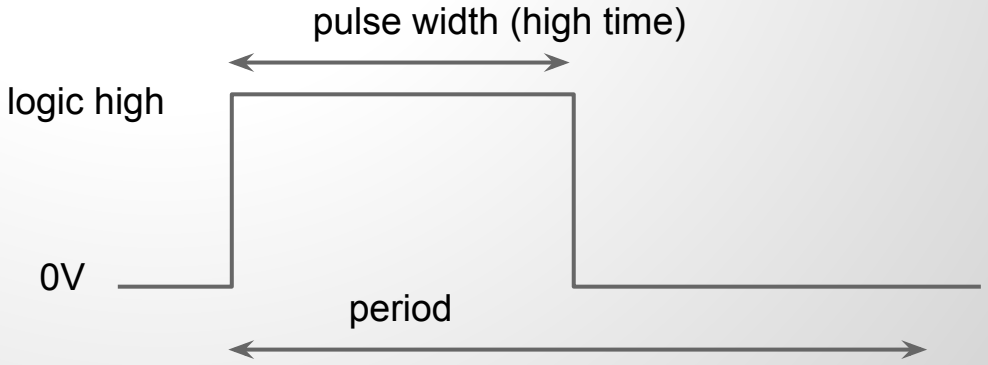


Control and calibration of servo motors



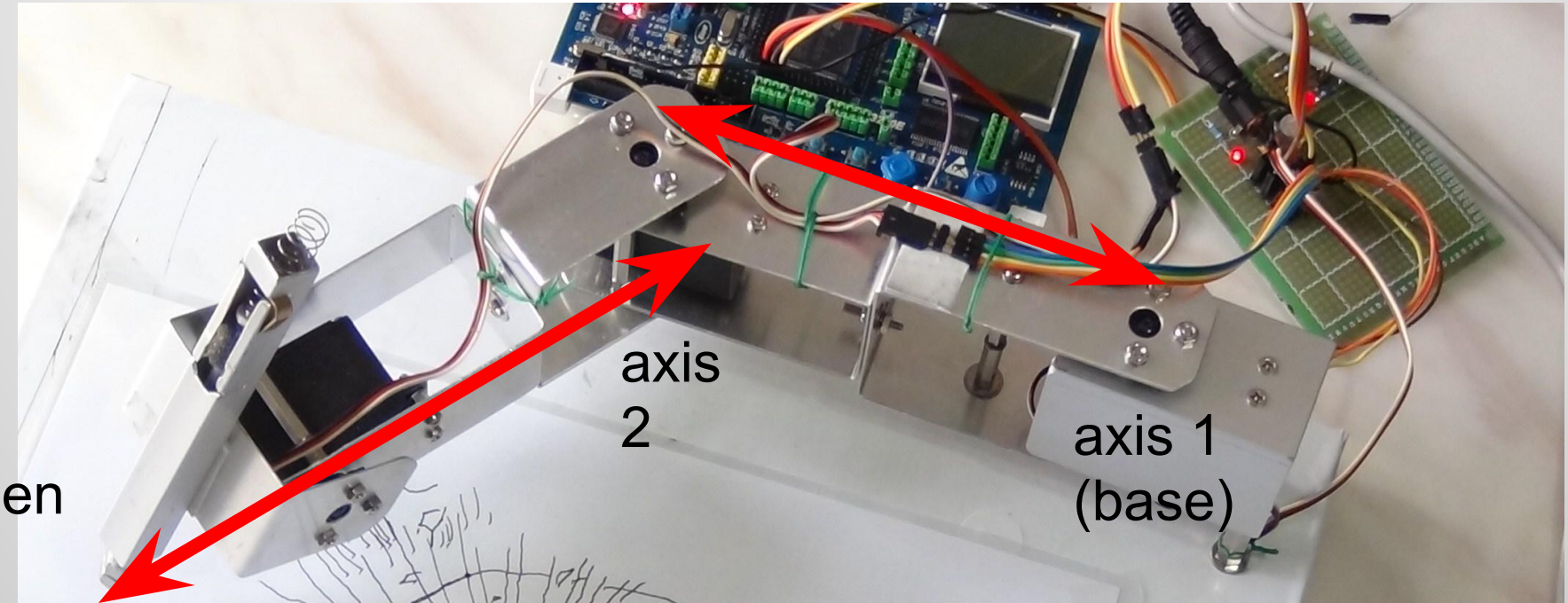
- pulse width determines turning angle
- typically servo motors signals at 50Hz
- STM32 specific:
  - system clock at 36MHz
  - prescaler = 10
  - period = 65535 = 0xFFFF
  - => output clock at 49.9Hz
  - high time from 0 to 65535
  - **fine pulse width control**

pulse width modulation (PWM)



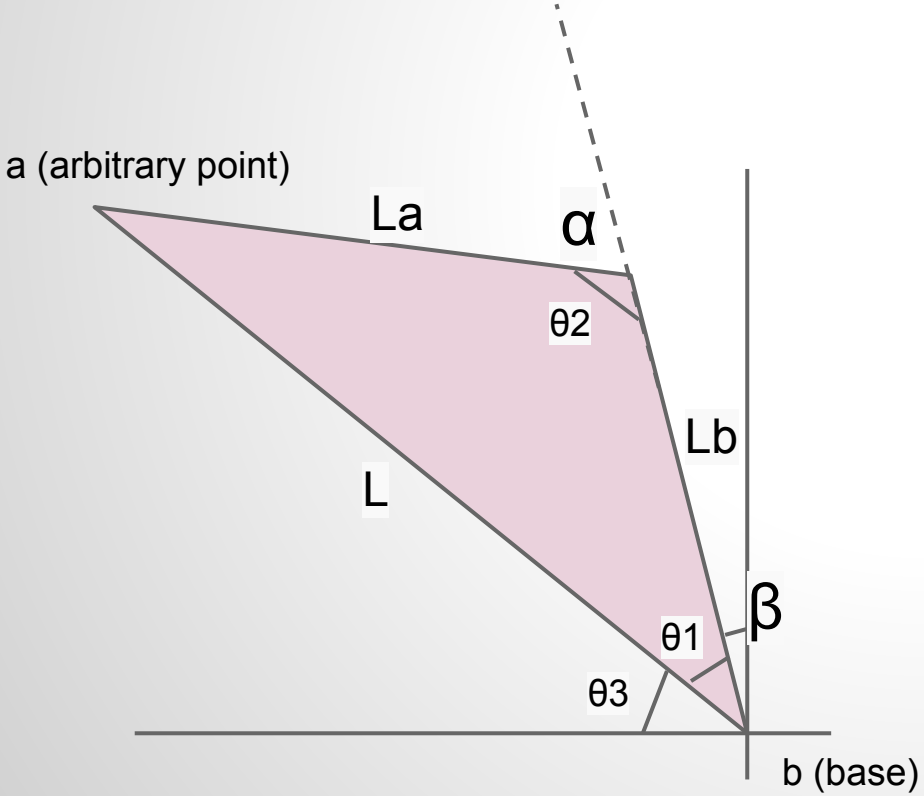
Calibration for arm

- axis1 to axis2: 140mm
- pen tip to axis2: 159mm
- all reflects in source code



Simple inverse kinematics

- not an isosceles triangle
- cosine law



$$cos\theta_1 = \frac{L^2 + Lb^2 - La^2}{2L(Lb)}$$

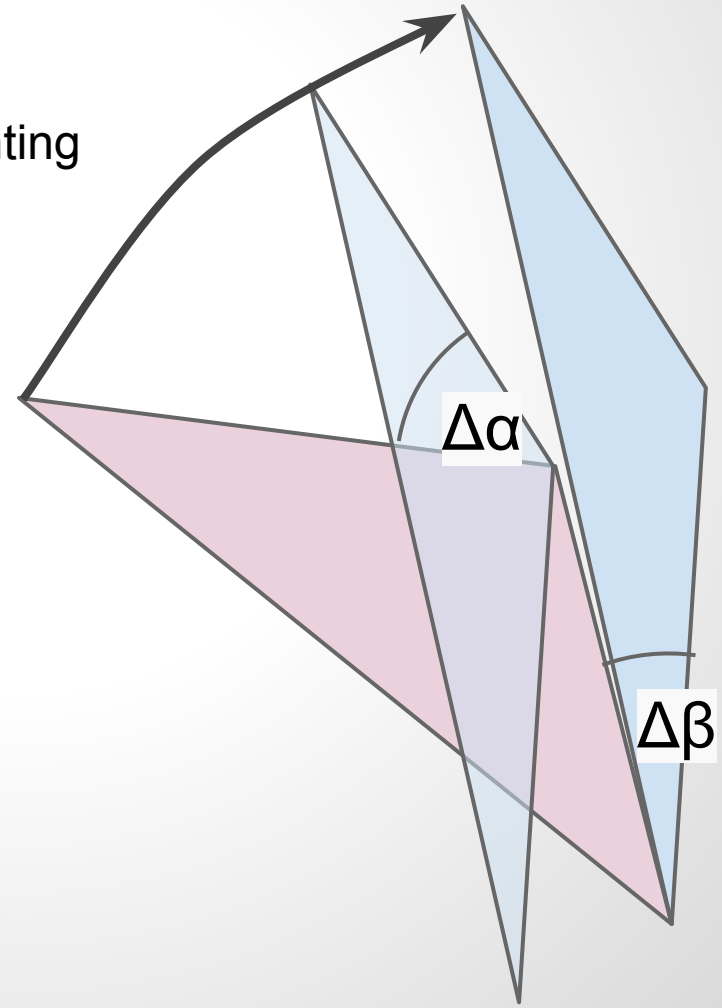
$$cos\theta_2 = \frac{La^2 + Lb^2 - L^2}{2La(Lb)}$$

Concurrent control

- 2 motors must turn concurrently
- such that they arrive at the same time
- each move at different speed!
- programmatically: C-style object orienting
- (curse of 1/0)

$$\Delta\theta = \omega \, \Delta t$$

$$\Delta\alpha / \Delta\beta = \omega\alpha / \omega\beta$$

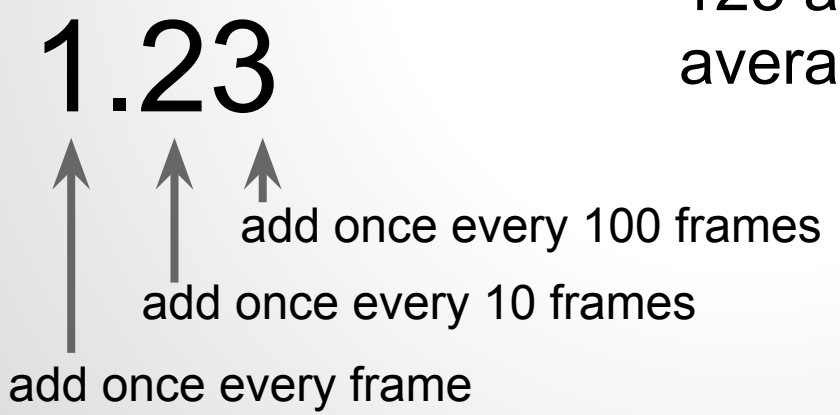


Even finer speed control?

- we divide 200 degrees into 8000 steps for precision
- if we move faster, we will lose precision
  - slower is finer
- 10 times less precise if we move at speed 10!
- say $\Delta\alpha / \Delta\beta = 1270/600 = 2.11666,,,$
- pulse width is integer!
  - $\omega\alpha / \omega\beta = 21/10 = 2.1$
    - fractional error build up over time
  - $\omega\alpha / \omega\beta = 127/60 = 2.11666,,,$
    - speed ratio is precise enough
    - motor movement is very coarse
- a hard-to-solve problem

the solution - time averaging

- use slowest speed possible
- slower motor at speed 1
- faster motor at "fractional" speed up to 2 decimal places
  - how? time averaging



123 additions in 100 frames
average: 1.23 every frame!

theoretic part end

demonstrate now!

Precisions

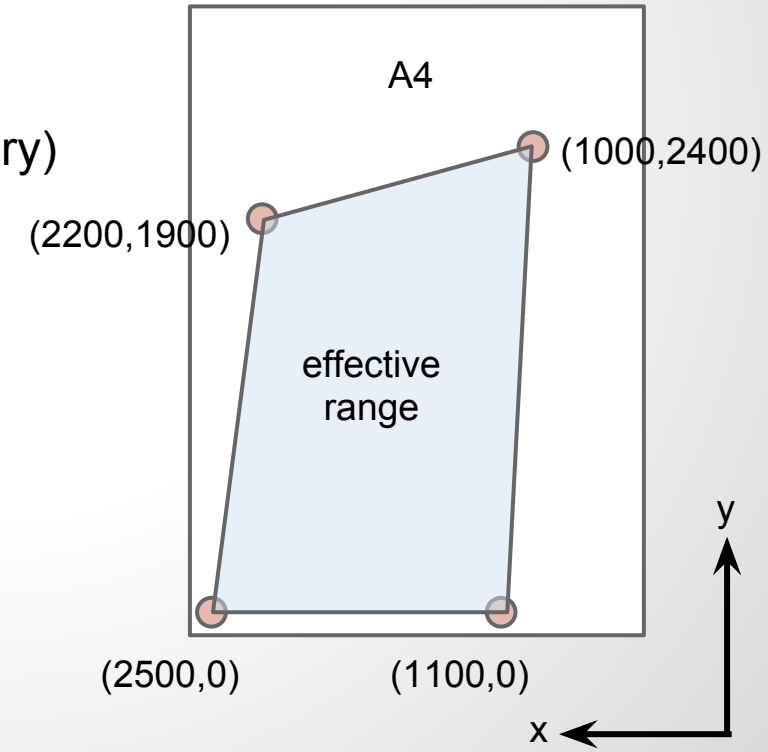
- angle at 1/100 degree

9000 -> 90 degrees

- rectangular coordinate at 1/10 mm

100 -> 10mm -> 1cm

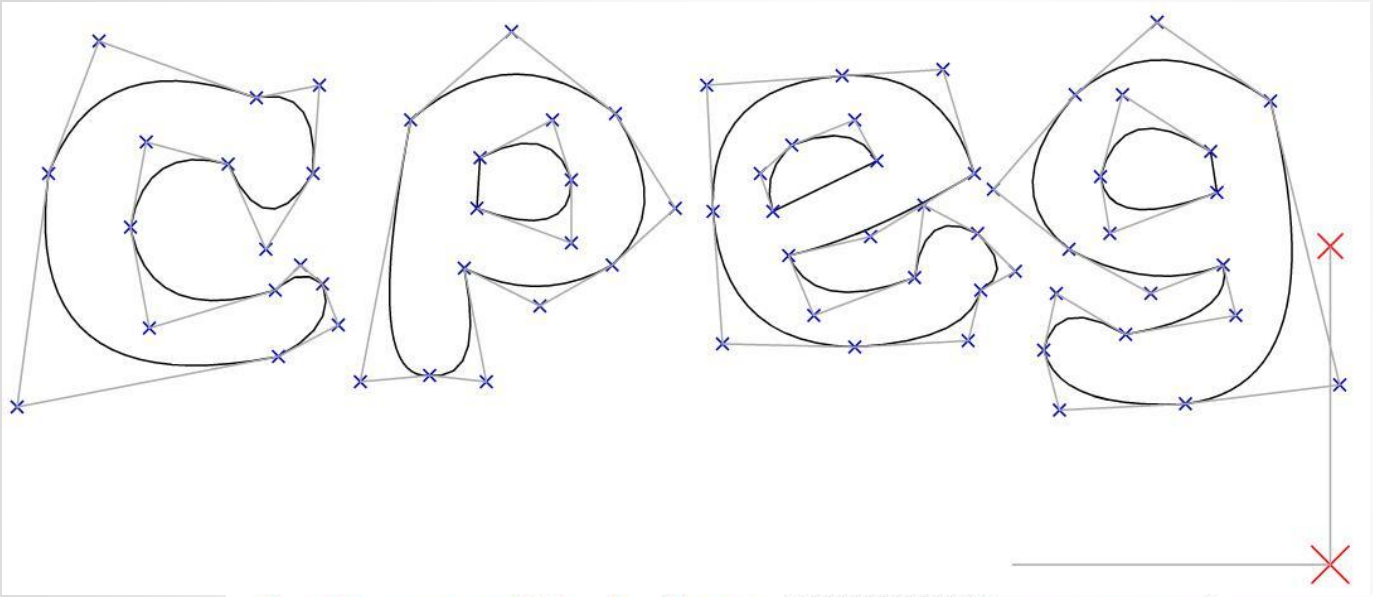
- (precision delivered by software, mechanical accuracy is another story)



implemented operations

```
{'A',9000,4500}, //turn base to 90 degree, axis to 45 degree
{'M',100,200}, //move to absolute coordinate (100,200)
{'L',300,400}, //pen down, draw a line to (300,400)
{'B',500,600}, //specify the control point of bezier curve
{'C',300,400}, //draw a degree 2 bezier curve to (300,400)
{'Q',0,0} //quit, indicating the last operation
{'a',1000,2000}, //turn 10 and 20 degree respectively
{'m',30,40}, //move relatively
{'l',50,60}, //draw a line relatively
{'b',50,50}, //control point relative to the current point of the plotter
{'c',100,150}, //bezier curve relative to the current point of the plotter
```

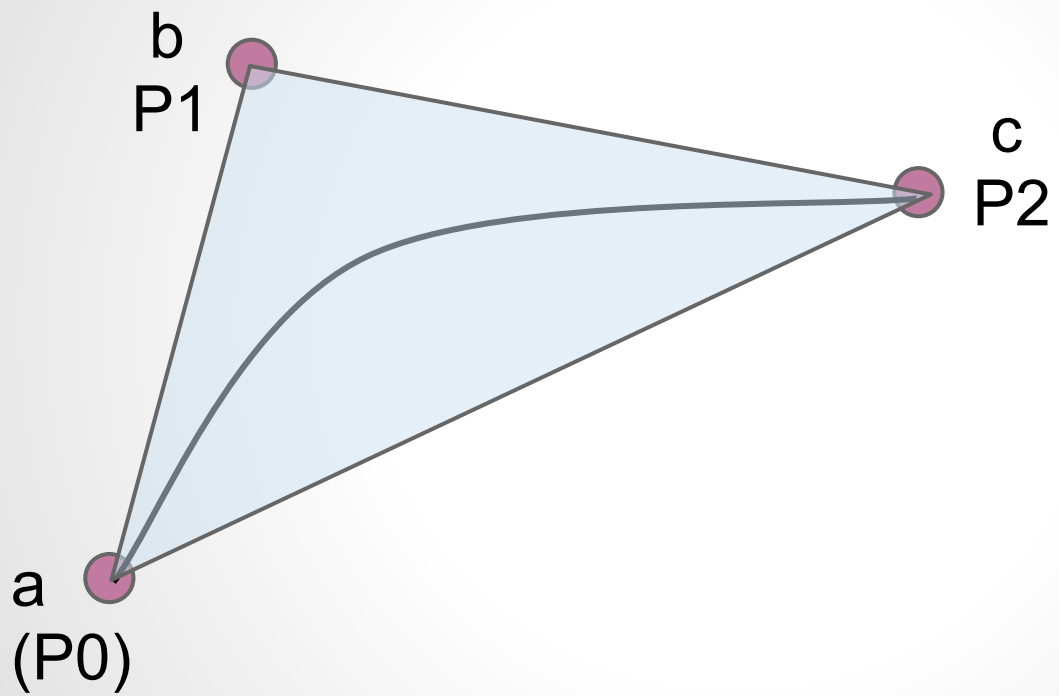
the designer software in companion



plotting graphic designer | change() | new bezier | new line | new dot | undo | redo


```
[{"x":206,"y":214},{"x":156,"y":196},{"x":140,"y":238},
{"x":140,"y":238},{"x":130,"y":312},{"x":213,"y":296},
{"x":213,"y":296},{"x":232,"y":286},{"x":227,"y":273},
{"x":227,"y":273},{"x":220,"y":267},{"x":212,"y":275},
{"x":212,"y":275},{"x":172,"y":287},{"x":166,"y":255},
{"x":166,"y":255},{"x":171,"y":228},{"x":197,"y":235},
{"x":197,"y":235},{"x":209,"y":262},{"x":224,"y":238},
{"x":224,"y":238},{"x":226,"y":210},{"x":206,"y":214},
{"x":255,"y":221},{"x":287,"y":193},{"x":320,"y":219},
```

(degree 2) Quadratic Bézier curves



$$\mathbf{B}(t) = (1-t)^2\mathbf{P}_0 + 2(1-t)t\mathbf{P}_1 + t^2\mathbf{P}_2 \ , t \in [0,1].$$