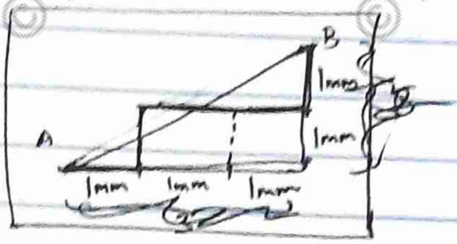


$(penX, penY)$;
$(usrX, usrY)$;

$\overline{V} = (\{usrX - penX\}, \{usrY - penY\})$
~~$= (S_x, S_y)$~~

define $S_x = |usrX - penX|$
& $S_y = |usrY - penY|$.
i.e. S_x & S_y are the no. of steps to cover to reach from (penX, penY) to (usrX, usrY).

To find: the delays b/w 2 consecutive pulses applied to the x & y motors so that the pen traces a straight line from A to B; i.e. ~~reach~~ depart from A & reach B in the same duration "t" following a ~~visibally~~ visiblally straight line path.

$t_{dx} \propto \frac{1}{x}$ & $t_{dy} \propto \frac{1}{y}$ ($\because$ as x increases, if a single move instruction is to take constant time, the distance should reduce delay b/w consecutive pulses accordingly.)

also, the move instruction starts & ends with a pulse. Delays are interleaved b/w consecutive pulses.

<move starts> - P — D — P — D — ····· — D — P — <move ends>

This implies the number of delays produced are 1 less than the number of pulses applied.

$\Rightarrow D_* = S_* - 1$ $\left[\begin{array}{l} D_x = S_x - 1 \\ D_y = S_y - 1 \end{array}\right]$

- Let T be the time to complete 1 move operation. Assume it to be configurable by the user programmer.

= ~~[illegible] Assume also that 1 pulse duration takes [illegible] time~~

~~Also~~ Thus total must add up to the time T = ~~Hence, the delay b/w [illegible] consecutive steps~~

~~$\Rightarrow$ [illegible] $t_{dx}(D_x-1)$~~

~~$\Rightarrow$ [illegible] $t_{dx} D_x = t_{dy} D_y$ or [illegible]~~

~~- The total delay = (delay b/w consecutive pulses) * (No. of times the delay occurred)~~

~~$$= T = t_{dx} \cdot (S_x - 1) = t_{dx} D_x$$~~

~~$$= t_{dy} \cdot (S_y - 1) = t_{dy} D_y.$$~~

~~$\therefore \boxed{t_{dx} D_x = t_{dy} D_y}$ $(D_x = S_x - 1 ;\ D_y = S_y - 1)$.~~

~~$\Rightarrow \boxed{\frac{t_{dx}}{t_{dy}} = \frac{D_y}{D_x} = \frac{S_y - 1}{S_x - 1}}$ & $t_{dx} = \frac{T}{D_x} \Rightarrow \boxed{t_{dx} = \frac{T}{S_x - 1}}$~~

~~& $t_{dy} = \frac{T}{D_y} \Rightarrow \boxed{t_{dy} = \frac{T}{S_y - 1}}$~~

- Let t_p be the time required to step ~~the~~ either motor in ~~any~~ either directions.

- Thus, we can say: $T = S_x t_p + \overbrace{(S_x - 1)}^{D_x} t_{dx} = S_y t_p + \overbrace{(S_y - 1)}^{D_y} t_{dy}$ where t_{dx} & t_{dy} are delays b/w consecutive X & Y pulses respectively.

- Thus, $\boxed{t_{dx} = \frac{T - S_x t_p}{S_x - 1}}$ & $\boxed{t_{dy} = \frac{T - S_y t_p}{S_y - 1}}$

- S_x & S_y are the total steps taken to reach from A_x to B_x & A_y to B_y respectively.

i.e. $S_x = \frac{B_x - A_x}{(\text{linear step distance})}$; $S_y = \frac{B_y - A_y}{(\text{linear step distance})}$

- 'linear step distance' can be calculated by:

declare as a macro or a global const in the config file

(i) assemble the plotter completely;

(ii) attach the pen to pen-end;

(iii) (a) write a program that rotates the X motor only, fully (360°), whilst keeping the pen down based on the stepping mode selected. Measure the marked line's length.

(b) Next, rotate the motor by 180° & measure the new line's length.

(c) Finally, rotate the motor by 90° & measure the last line's length.

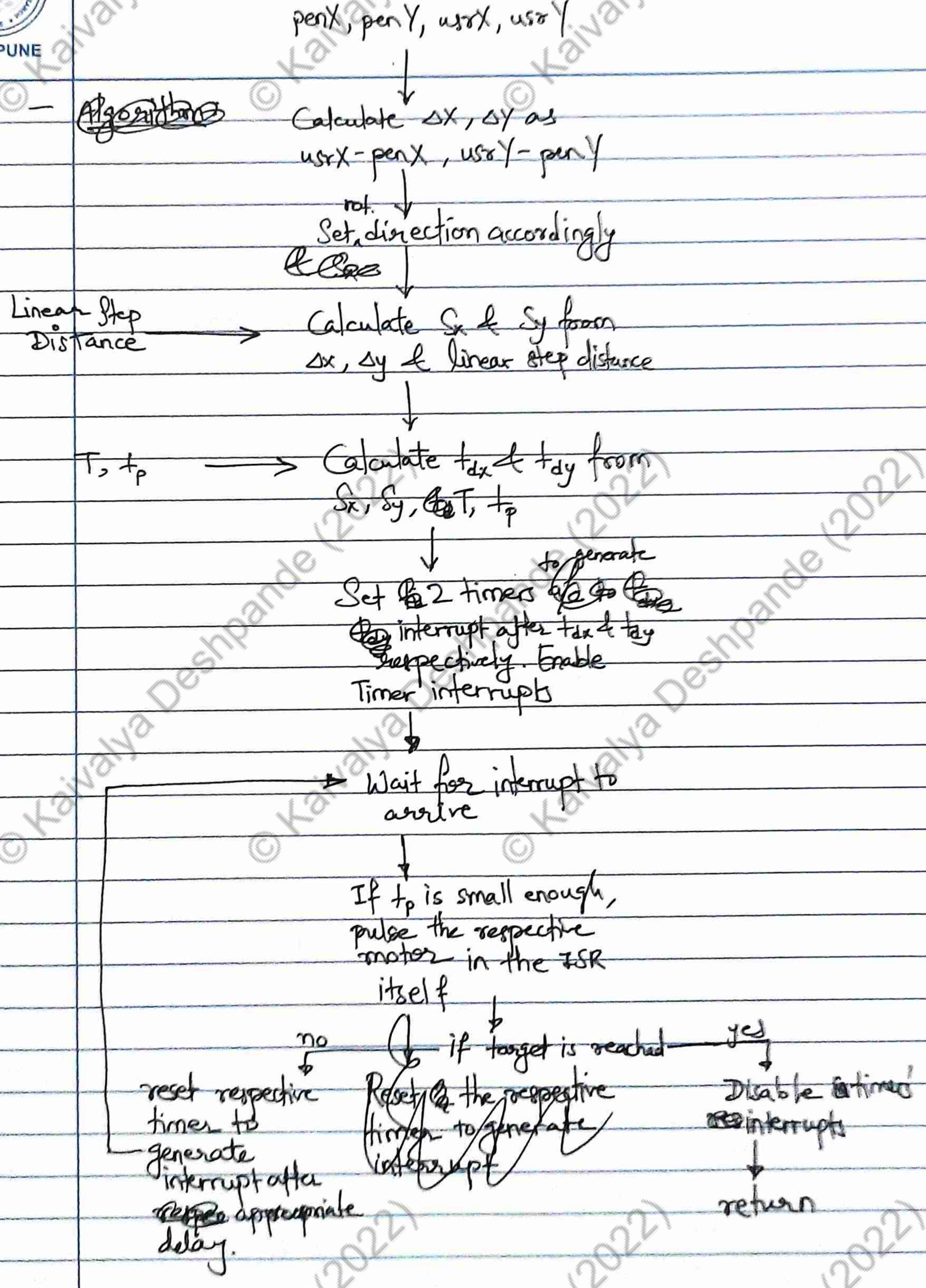
$$\text{Find Linear step Distance (a)} = \frac{\text{measured distance in (a)}}{(\text{steps for 1 complete rotation})\ 200}$$

$$\text{Linear step Distance (b)} = \frac{\text{measured distance in (b)}}{(\text{steps for } \tfrac{1}{2} \text{ rot.})\ 100}$$

$$\text{Linear step Distance (c)} = \frac{\text{measured distance (c)}}{(\text{steps for quarter rot.})\ 50}$$

In the end, $\boxed{\text{Linear step Distance} = \frac{\sum_{i=a}^{c} \text{Linear step Distance } (i)}{3}}$

(one can take > 3 readings for greater accuracy).


Algorithm
penX, penY, usrX, usrY
Calculate ΔX, ΔY as usrX - penX, usrY - penY
Set rot. direction accordingly
Linear Step Distance
Calculate Sx & Sy from Δx, Δy & linear step distance
T, tp
Calculate tdx & tdy from Sx, Sy, T, tp
Set 2 timers to generate interrupt after tdx & tdy respectively. Enable Timer interrupts
Wait for interrupt to arrive
If tp is small enough, pulse the respective motor in the ISR itself
if target is reached
no
yes
reset respective timer to generate interrupt after appropriate delay.
~~Reset the respective timer to generate interrupt~~
Disable timer interrupts
return