E.18) i)

```
                    ( Start )
                        │
                        ▼
           ┌──→─────────┐
           │            │
           │       ╱◇╲        Yes      ( End in )
           │     ╱ Reached ╲──────────( success )
           │     ╲ Goal   ╱
           │       ╲◇╱
           │         │
           │        No
           │         │
           │         ▼
           │     ╱◇◇◇╲      Yes       ⎛ End in    ⎞
           │   ╱ dist. from ╲──────────⎜ failure   ⎟
           │   ╲ current-position      ⎜ "Obstacle in⎟
           │   ╱ to closest ╲          ⎜ between    ⎟
           │   ╲ polygon    ╱          ⎝ start & goal⎠
           │     ╲ < step ╱
           │       ╲◇◇╱
           │         │
           │        No
           │         │
           │         ▼
           │   ┌───────────┐
           └───┤ take step │
               │ toward    │
               │ the goal  │
               └───────────┘
```
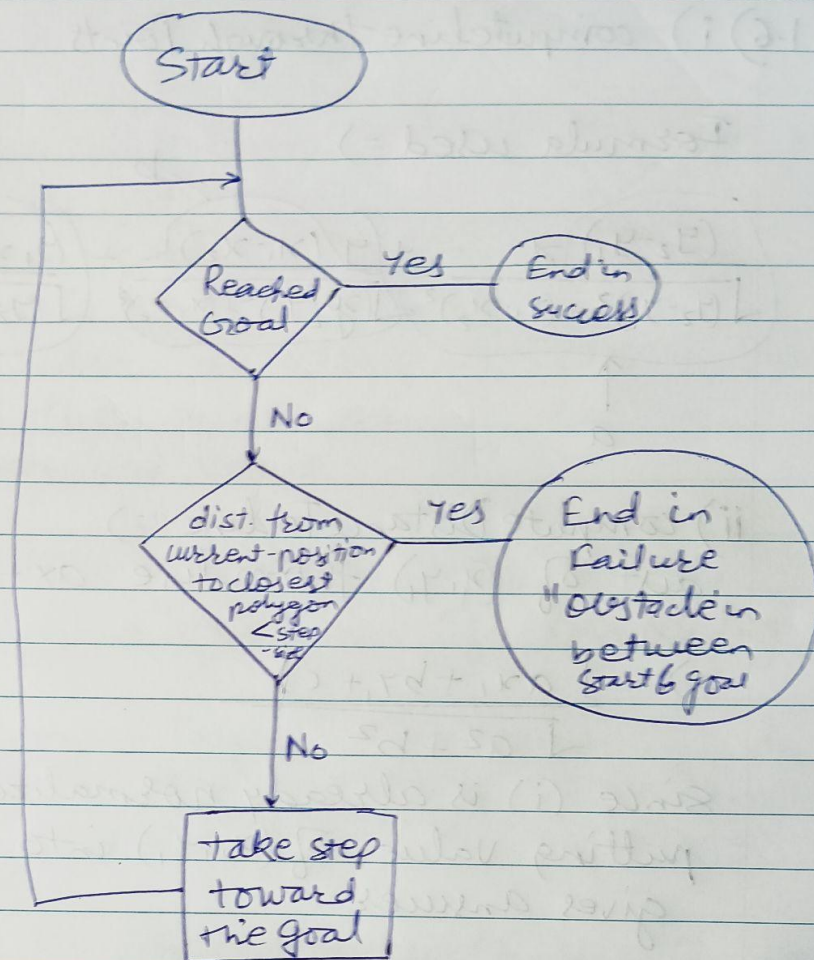
ii) I will replace line 6. I will ask robot
to circumnavigate the obstacle and record
distance from robot to goal at each
instance. Once circumnavigation is done,
the robot will be asked to step toward
the goal from the point with min dist.
from the goal.

functions =) computeTangentToPolygon ⇒ to get
tangent once obstacle is hit.
computeDistanceToPolygon ⇒ to get if
obstacle is hit or not.

E1.6) i) compute line through Points

Formula used =)

$$\underbrace{\frac{(y_2-y_1)x}{\sqrt{(y_2-y_1)^2+(x_1-x_2)^2}}}_{a} + \overbrace{\frac{y(x_1-x_2)}{\sqrt{(y_2-y_1)^2+(x_1-x_2)^2}}}^{b} + \overbrace{\frac{(y_1x_2-x_1y_2)}{\sqrt{(y_2-y_1)^2+(x_1-x_2)^2}}}^{c} = 0$$
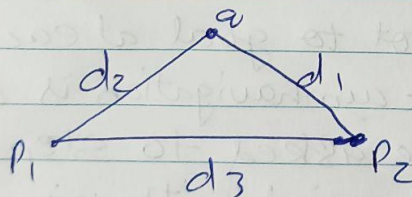
ii) compute Distance To line =)
  dist of (x_1, y_1) from line $ax+by+c=0$.

$$=) \quad \frac{|ax_1+by_1+c|}{\sqrt{a^2+b^2}}$$

Since (i) is already normalized, directly putting value of (x_1, y_1) into eq^n of (i) gives answer

iii) compute Distance Point To Segment =)
  Calculate 3 distance

if $d_3^2 > d_2^2 + d_1^2$
  $\longrightarrow w = 0$

if $d_2^2 > d_1^2 + d_3^2$
  $\longrightarrow w = 2$

if $d_1^2 > d_2^2 + d_3^2$
  $\longrightarrow w = 1$

Simple application euclidean geometry

E 1.7) i) Compute Distance Point To Polygon
use compute Distance Point To Segment on
each line segment of Polygon to get.

$$\begin{bmatrix} d_1 & d_2 & \cdots & d_n \\ w_1 & w_2 & & w_n \end{bmatrix}$$

for all line segments.
consider distances d_i only if $w_i = 0$,
and then find min of d_is to
required value,

if there is not d_i with $w_i = 0$.
then find the min of remaining d_j's.

ii) Compute Tangent Vector of Polygon
Compute distance of Point from each
segment and Point.

Consider segments with $w = 0$. only.
find min of those. And thus the
point on the polygon which is closest.
~~Find perpendicular to the line~~
If point is on segment use it's slope
for tangent.

Only the bug_base algorithm has been implemented. The implementation was found to be successful, the robot travels in the direction of the goal and stops when encounters the obstacle.

```
ashay@ashay:~/catkin_ws$ rosrun sc627_assignments bug_base.py
0 0 5 3
[[1, 3, 1], [0, 0, 2]]
[[2, 4, 5], [3, 1, 2]]
0.062034469382875386 0.03341577597781474 0.577266785364406
0.12369747539598572 0.07526937299059702 0.6271303672434915
0.18237939981818177 0.11457639162661323 0.5890711936496823
0.2442855540340817 0.14849135768449231 0.5429063441707744
0.30639221898859526 0.18355391787836367 0.5547257682446229
0.36897548893185245 0.2187999678117878 0.5347496230631534
0.431658805111559 0.2536373467147558 0.5570686963986332
0.49244365420131014 0.2946713415756893 0.4929754817100988
0.552292261171283 0.33289147594893276 0.5774404819180732
0.6139599436303053 0.36761603549334226 0.5180552341337021
0.6759975224464925 0.4026874171900638 0.5134416261587722
0.7373950208813803 0.43801800991761325 0.5249054139401206
0.7964539698347166 0.47585732963880806 0.5650057983377046
0.8586837092708411 0.5114597100750352 0.5354532021420425
0.9198727217176443 0.5469504983423139 0.5217533522916324
0.9198727217176443 0.5469504983423139 0.5217533522916324
Obstacle encountered stopping now
ashay@ashay:~/catkin_ws$
```