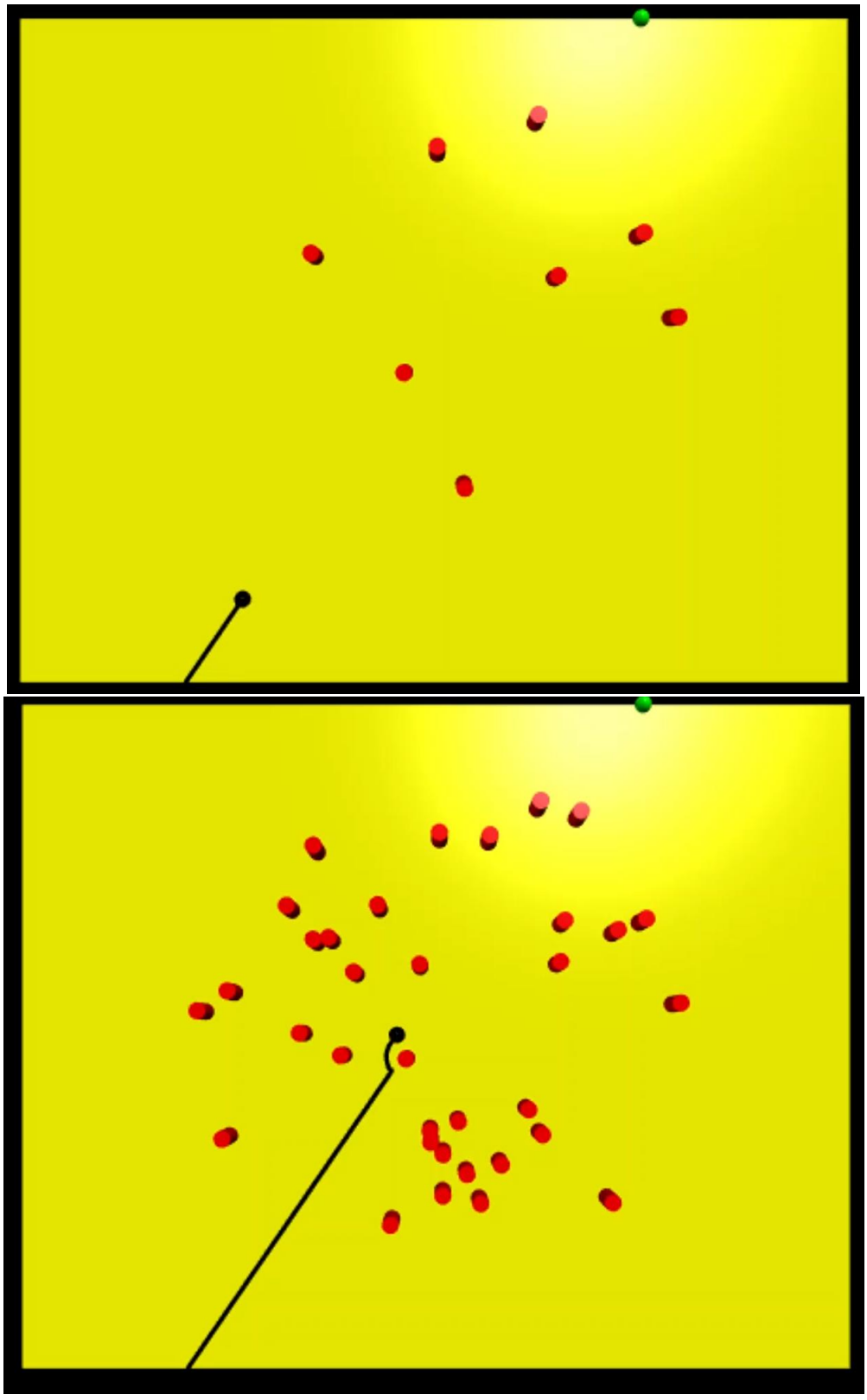
Assignment 4

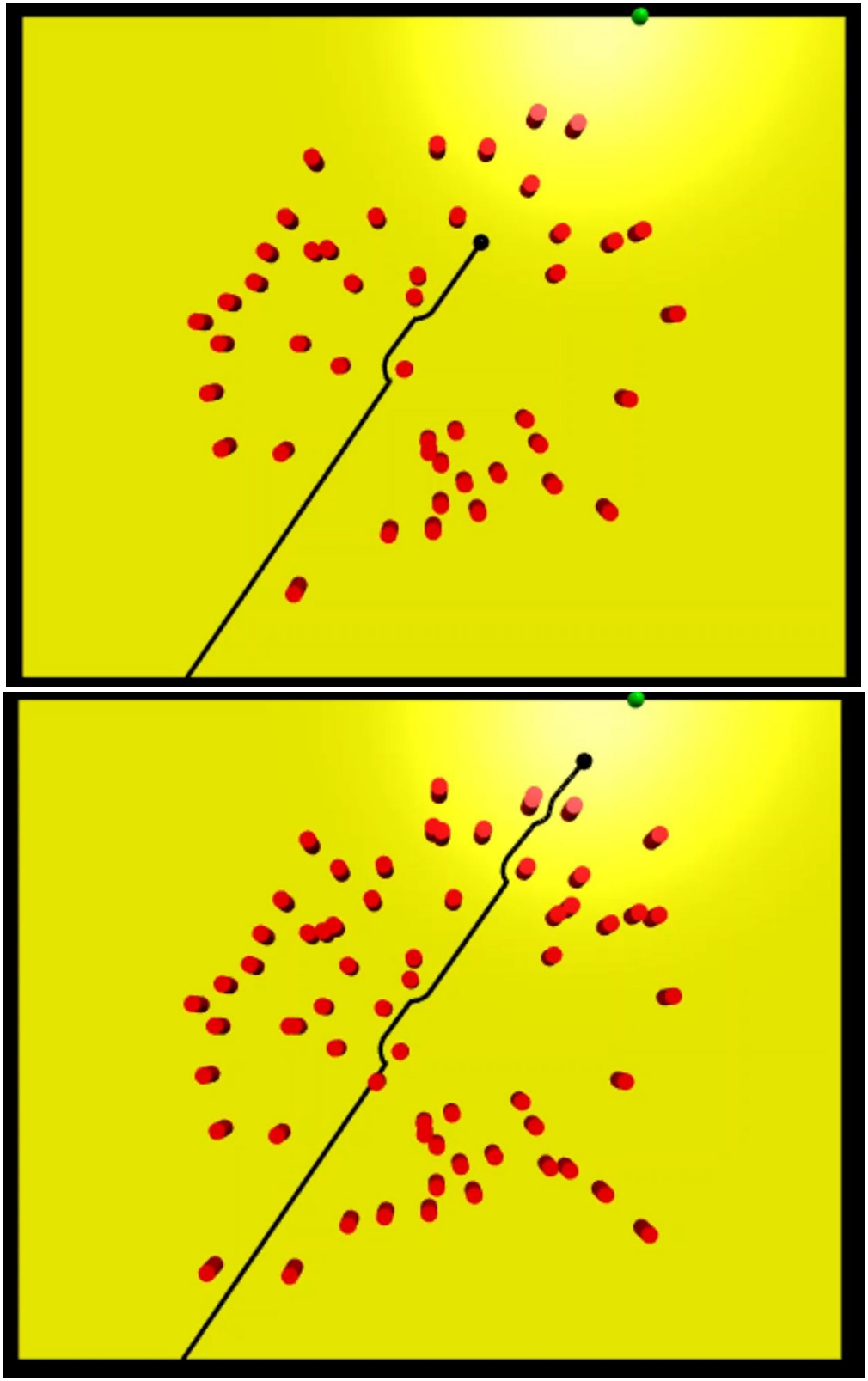
Fahad Alsuliman

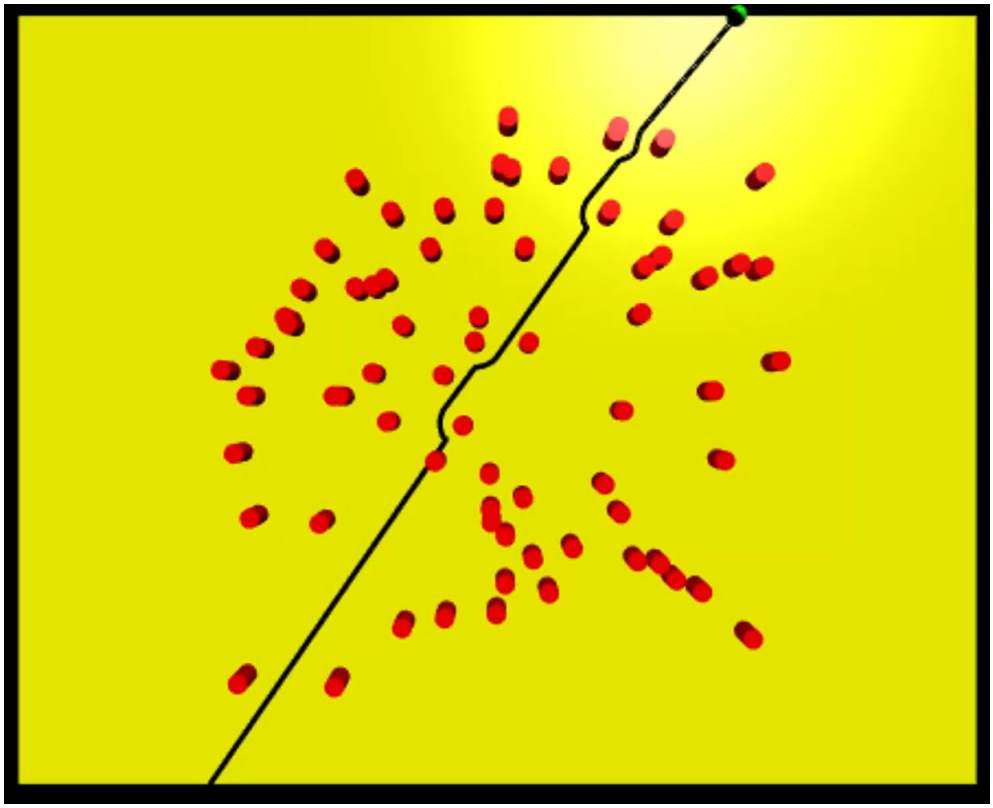
Include :

1- Preview
2- Code
3- Animation will be attached in canvas

Preview







Untitled3

March 6, 2020

[ ]:
```
#importing the required libraries
import math
from vpython import *
import random
```

[ ]:
```
#the field cost calcuation , it has 3 situations
# if the distance is greater than force field return 0
# if distance is less than or equal force field return the function
# if distance is so close to the object return the function
def fieldValueCalcuation(position,nObstacle):
    distance = (position - nObstacle[0]).mag
    if(distance >  25 ):
        return 0
    elif(distance <= 25 and distance >15):
        fieldValue = math.log(25/distance)*30
    elif(distance<= 15):
        fieldValue = math.log(15/distance)*30
    return fieldValue
```

[ ]:
```
def minimum_cost_function(directionSet,totalObstacles,dt,currentposition,goal):
    totalcost = []
    #loop through each direction (x,-x,y,-y)
    for d in directionSet:
        fieldcost = 0
        counter = 0
        suggestedStep = currentposition + d
        #cacluate the distance between goal and suggested step
        curcost = sqrt(((suggestedStep.x - goal.x)**2) + (( suggestedStep.y -␣
 ↪goal.y)**2))
        # calcuate field value for each obstacle
        for each in totalObstacles:
            counter = counter + 1
            fieldcost = fieldcost + fieldValueCalcuation(suggestedStep,each)
        curcost = curcost + fieldcost
        totalcost.append(curcost)
# gradient descent
    dx = totalcost[0]-totalcost[1]
```

```
    dy = totalcost[2]-totalcost[3]
    x =  vector(dx , dy, 0)
    # find new location
    newPosition = currentposition - x.norm()*dt
    return newPosition
```

[ ]:
```
#draw the objects
directionSet = [vector(2, 0, 0), vector(-2, 0, 0), vector(0, 2, 0), vector(0,␣
↪-2, 0)]
totalObstacles = []
box = box(pos=vector(0, 0, 0), length=1000, height=800, width=0.1, color=color.
↪yellow)
startingPoint = sphere(pos=vector(-300, -400, 0), radius=10, color=color.black␣
↪, make_trail = True)
goalPoint = sphere(pos=vector(250, 400, 0), radius=10, color=color.green )
goal = vector(250, 400, 0)
dt = 2
i = 1
```

[ ]:
```
# as long as the program didnt reach the target
# it should calcuate the next postion
# and add an obstacle every 8 steps
while ((startingPoint.pos - goal).mag >3):
    rate(50)
    startingPoint.pos = minimum_cost_function(directionSet, totalObstacles, dt,␣
↪startingPoint.pos, goal)
    i = i + 1
    if ( i % 8 == 1):
        temp5 = cylinder(pos=vector(int(random.uniform(-300, 300)), int(random.
↪uniform(-300, 300)), 0),
                         axis=vector(0, 0, 30), radius=10, color=color.red)
        totalObstacles.append([temp5.pos, temp5.radius])
        i = i+1
```