Random Walk

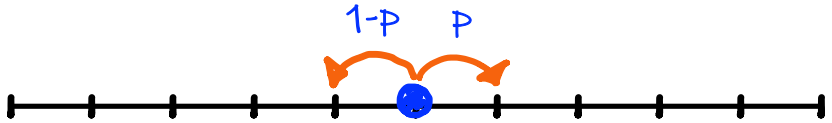
IKN

Random walk

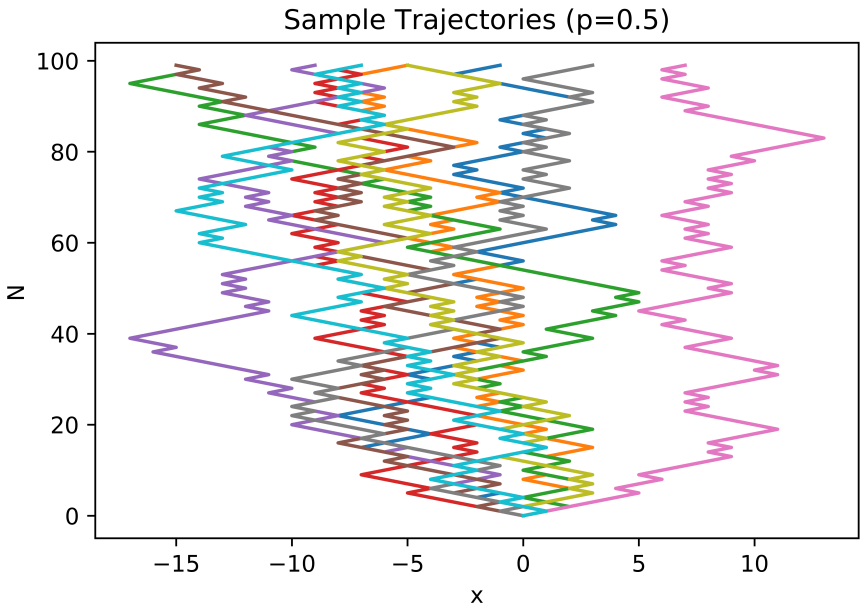
- A process, a model or a rule to generate path sequence of random motion.
- Natural phenomena can be modelled as random walk:
  - The path traced by a molecule as it travels in a liquid of a gas,
  - The search path of foraging animal,
  - The price of fluctuating stock,
  - Investment theory of stock market, etc.

Random Walk in Infinite Space

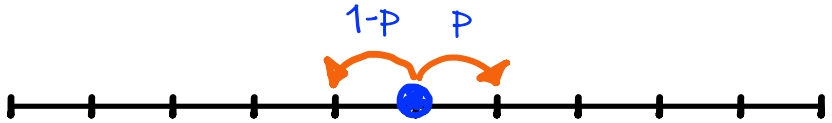
Random walk: 1D



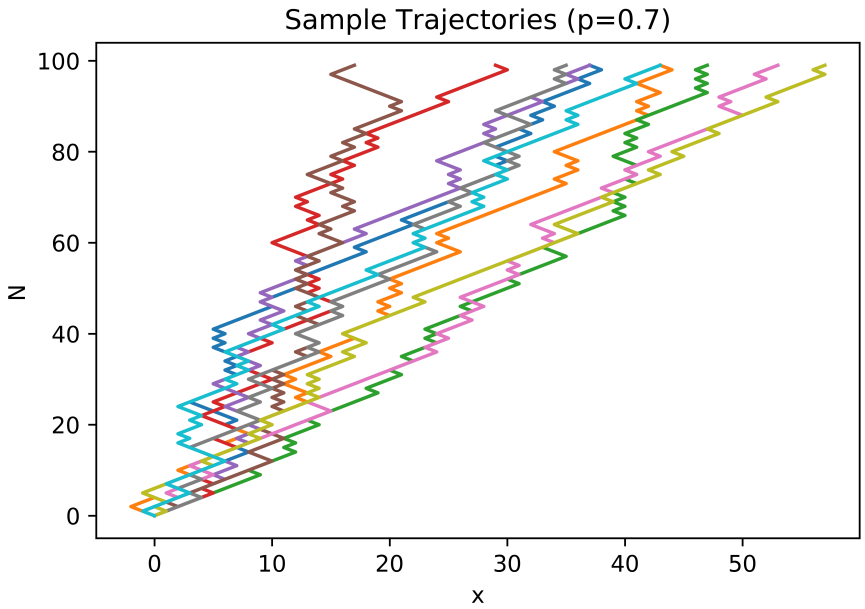
At each step particle jumps to the right with probability p and to the left with probability 1-p



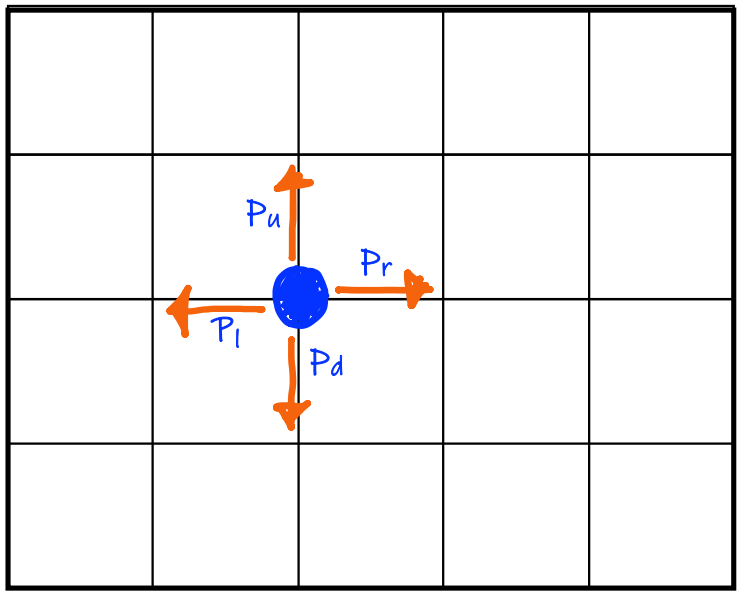
Random walk: 1D

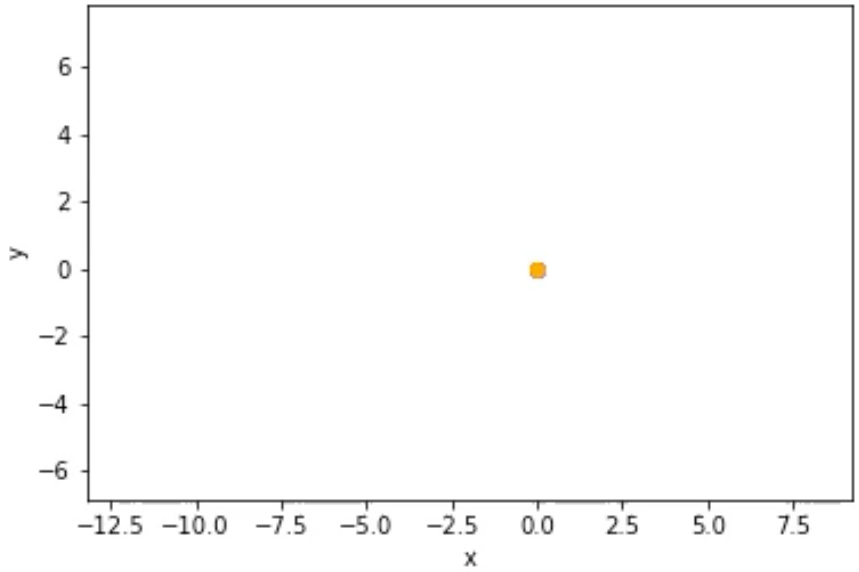


At each step particle jumps to the right with probability p and to the left with probability 1-p



Random walk: 2D (4 direction)

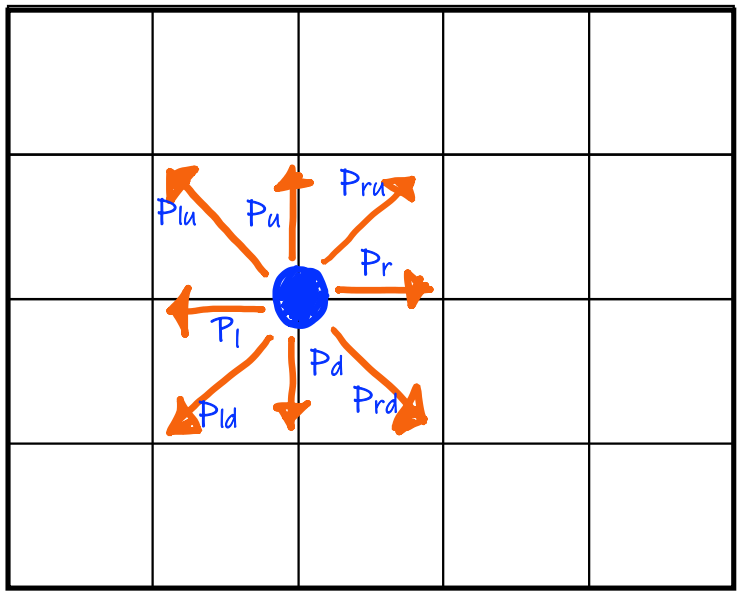


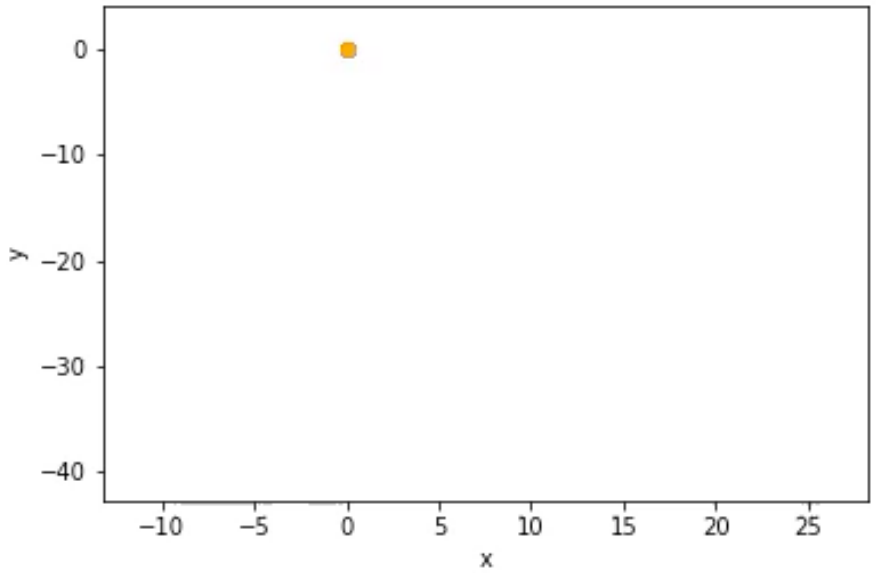


In the case similar of probability,

$P_r = P_l = P_d = P_u = 0.25$

Random walk: 2D (8 direction)

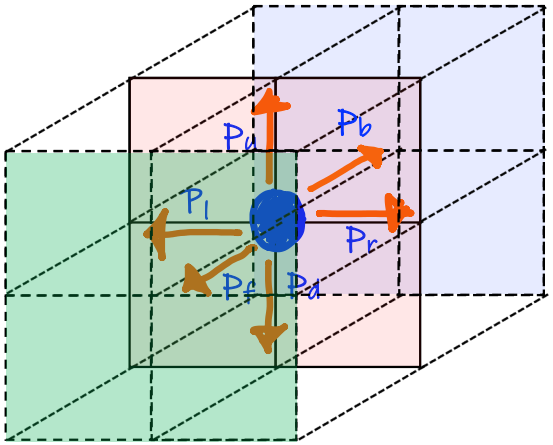




In the case similar of probability,

$P_r = P_l = P_d = P_u = P_{ru} = P_{lu} = P_{rd} = P_{ld} = 0.125$

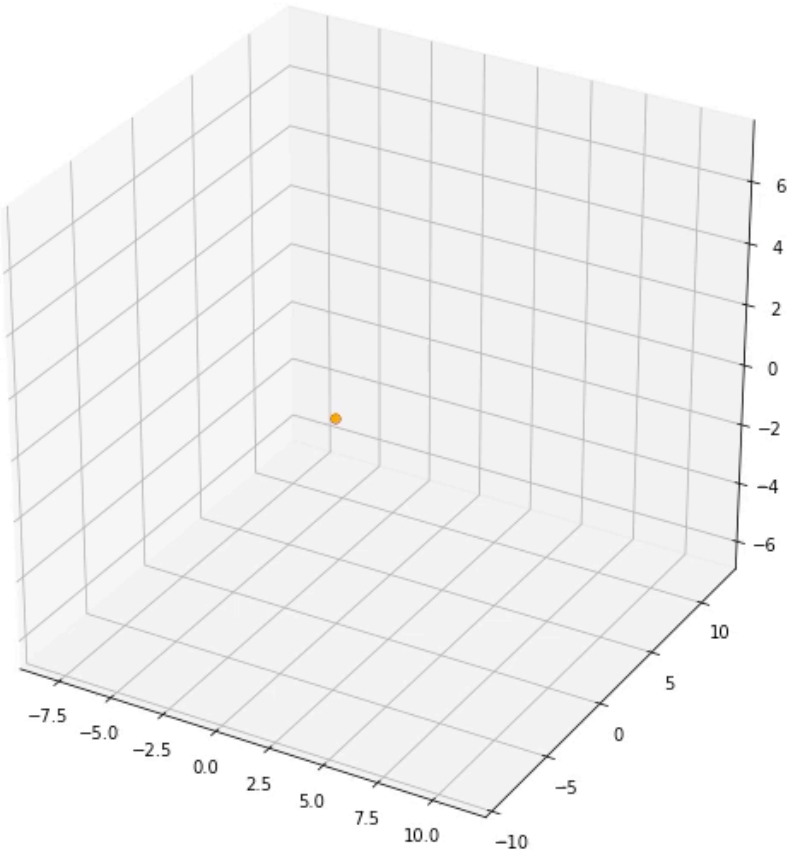
Random walk: 3D (6 direction)



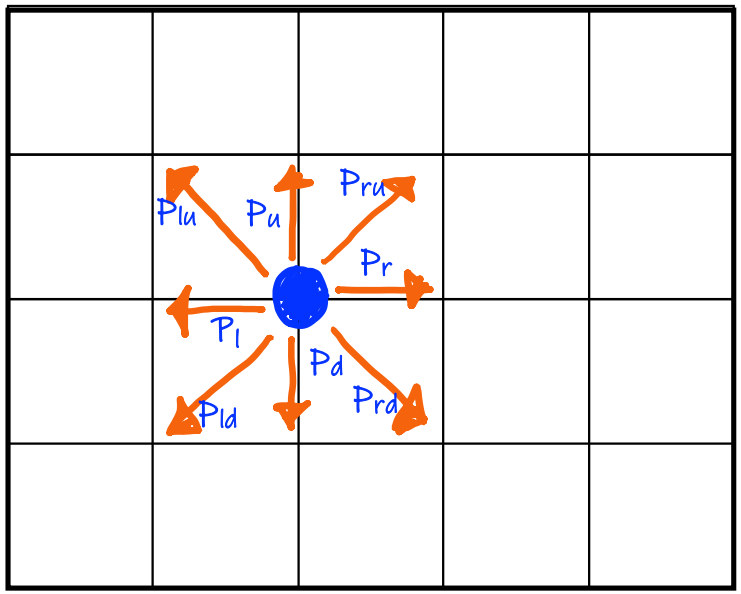
In the case similar of probability,

$P_r = P_l = P_d = P_u = P_f = P_b = 0.20$

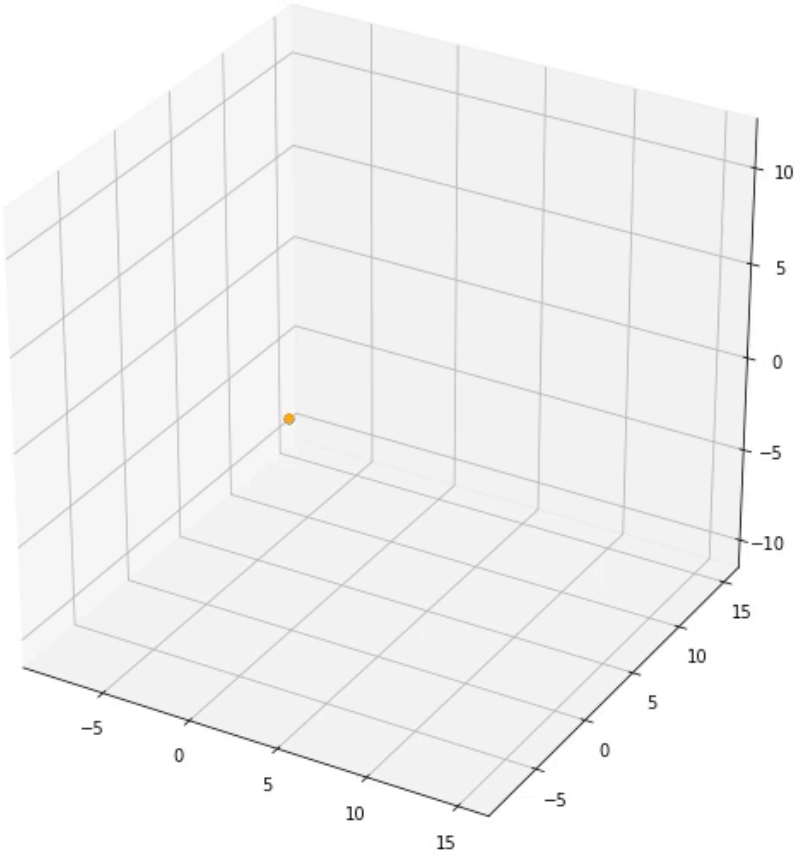
(with total range of probability = 1.20)



Random walk: 3D (24 direction)

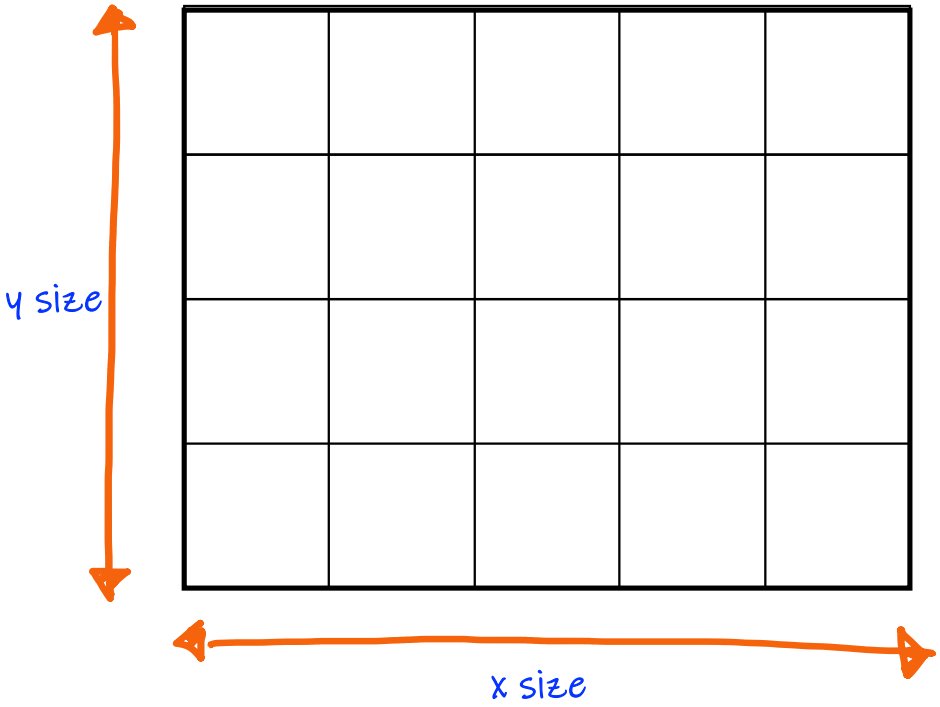


8 direction for each x, y, and z axis



Random Walk in Finite Space

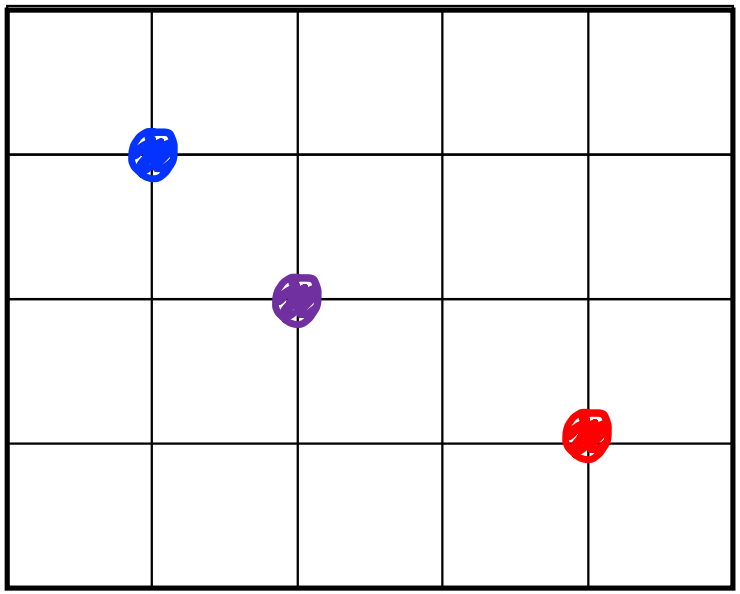
Defining size of simulation space




```
# define simulation space
x_min      # minimum value of x
x_max      # maximum value of x
y_min      # minimum value of y
y_max      # maximum value of y

x_range = x_max – x_min
y_range = y_max – y_min
```

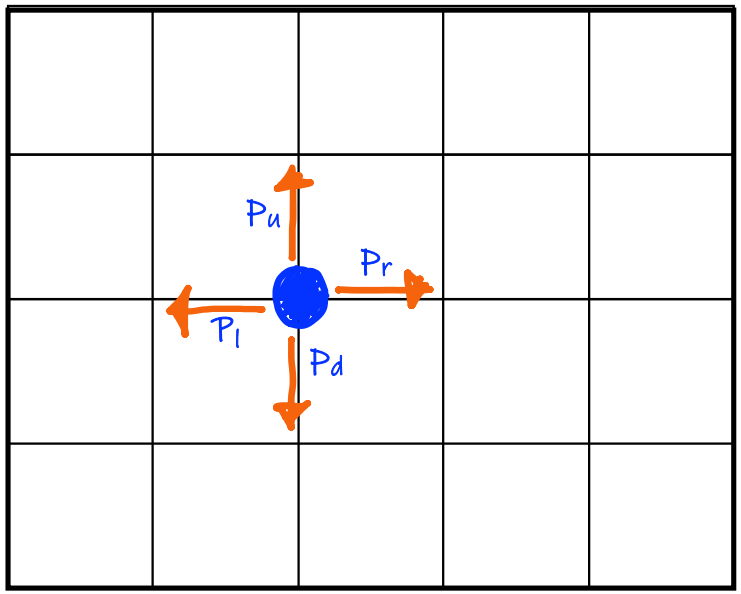
Defining initial position



```
# define initial position
n_particle      # number of particle

for i = 1:n_particle
    x_pos_i = [x_min ~ x_max]  # x position of i^th particle
    y_pos_i = [y_min ~ y_max]  # y position of i^th particle
```

Update position: Random walk 2D (4 direction)

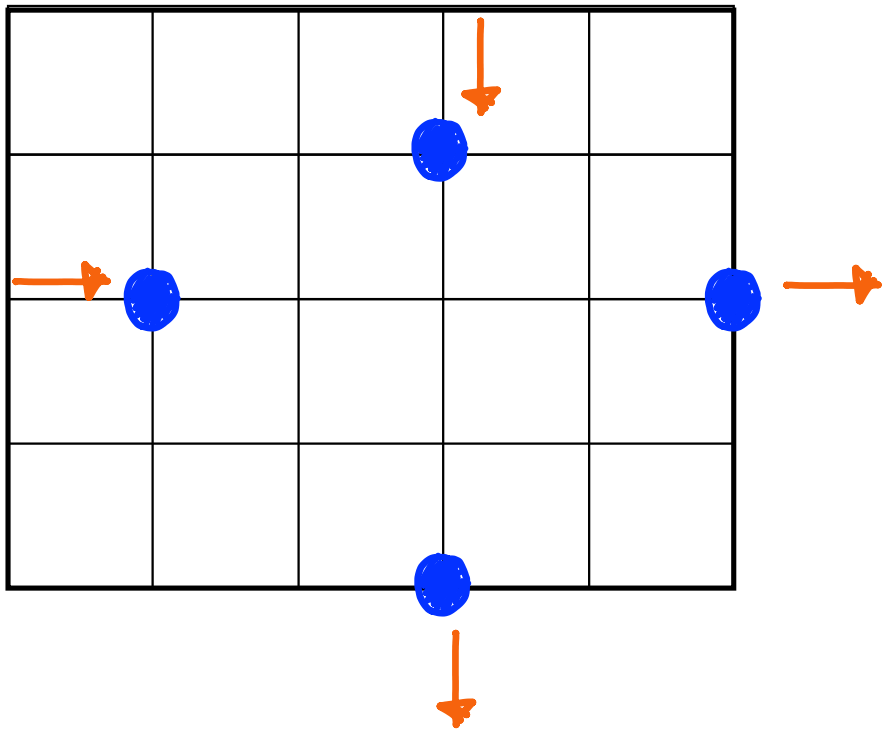


In the case similar of probability,

$P_r = P_l = P_d = P_u = 0.25$

```
# update position
rand = [0 ~ 1]      # random number
# right
if rand <= 0. 25:
    x = x + 1
# down
else if rand <= 0.50:
    y = y - 1
# left
else if rand <= 0.75:
    x = x - 1
# up
else :
    y = y + 1
```

Periodic boundary condition (PBC)



```
# periodic boundary condition
# correction of x axis
if x > x_max:
    x = x - x_range
if x < x_min:
    x = x + x_range
# correction of y axis
if y > y_max:
    y = y - y_range
if y < y_min:
    y = y + y_range
```

Pseudocode: Random walk 2D 4 direction

```
# initialization
n_particle     # number of particle
n_iter         # number of iteration
x_pos          # initial position of x
y_pos          # initial position of y

# iteration
for i = 1:n_iter
      for j = 1:n_particle
            # generate rand
            rand()          # random number
            # update x and y position based on
            # the defined probability
            x_pos[n,i+1] = x_pos[n,i] + dx
            y_pos [n,i+1] = x_pos [n,i] + dy
            # perform pbc correction
            store new position
```

```
# update position
rand = [0 ~ 1]      # random number
# right
if rand <= 0. 25:
    x = x + 1
# down
else if rand <= 0.50:
    y = y - 1
# left
else if rand <= 0.75:
    x = x - 1
# up
else :
    y = y + 1
```

Pseudocode: Random walk 2D 4 direction