AML_Lab3_Group8

Hyungyum Kim, Milda Poceviciute, Boxi Zhang, Fanny Karelius

2018 - 10 - 8

Question 1

A Hidden Markov model is implemented with the following features:

Transition model:

$$p(z_t|z_{t-1}) = \frac{1}{3}[N(z_t|z_{t-1},1) + N(z_t|z_{t-1}+1,1) + N(z_t|z_{t-1}+2,1)]$$

Emission model:

$$p(x_t|z_t) = \frac{1}{3}[N(x_t|z_t,1) + N(x_t|z_t+1,1) + N(x_t|z_t-1,1)]$$

Initial model:

$$p(z_1) = Uniform(0,100)$$

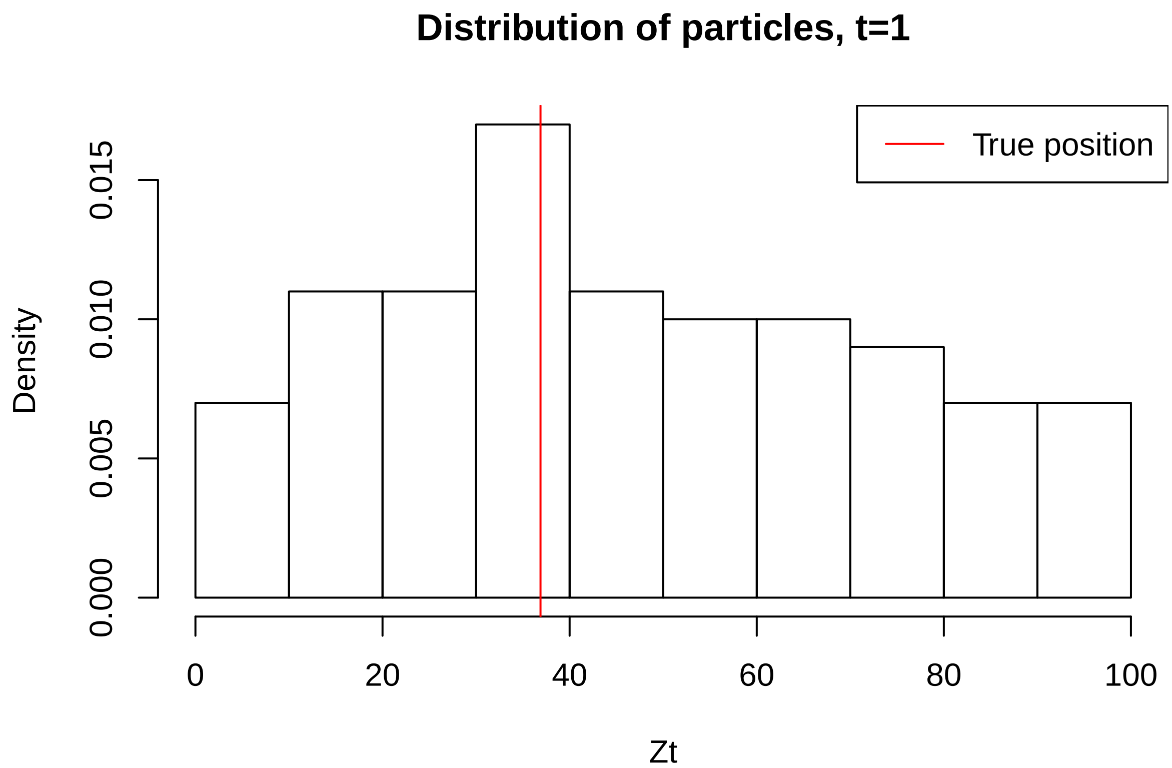
The functions are as follows:

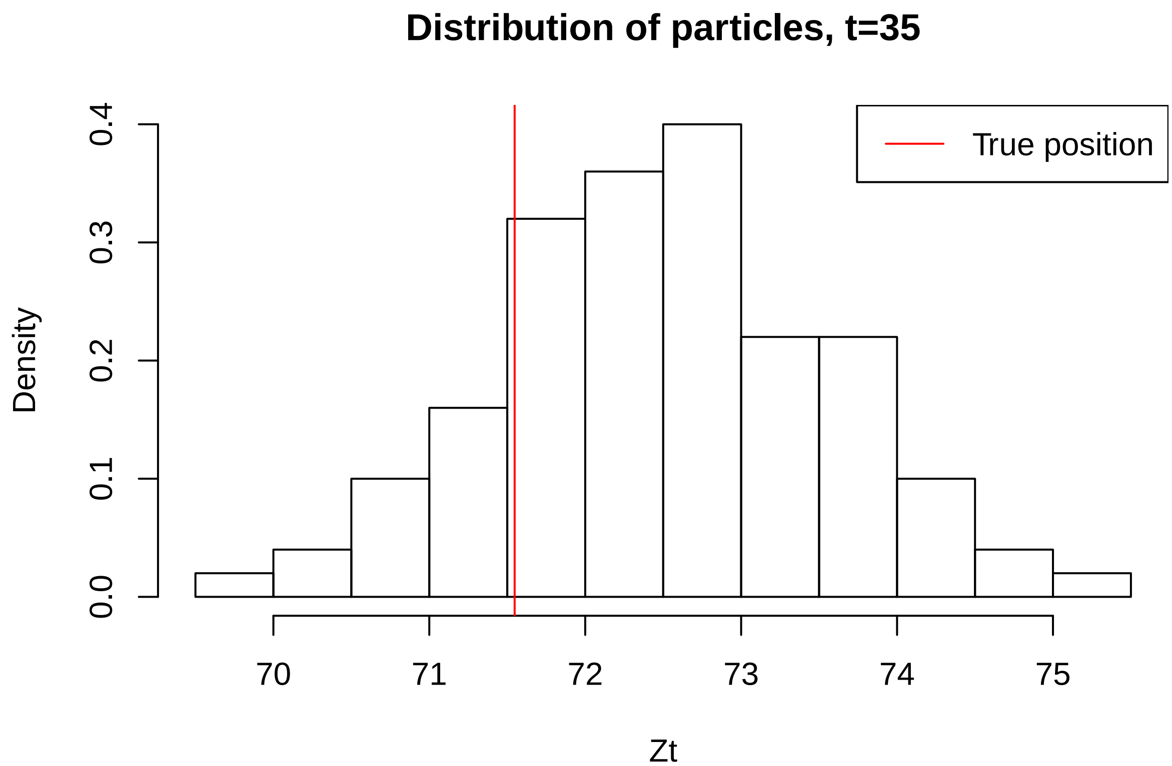
```r
# M - number of particals
# Iter - number of time steps T
# z is latent variable
# x is the observation

# Function for sampling from a mixture model
sampleMix <- function(mean1,mean2,mean3,sdX){
   u <- sample(1:3,1)
   means <- c(mean1,mean2,mean3)
   return(rnorm(1,mean=means[u],sd=sdX))
}

## Function that creates the SSM model
create_SSM <- function(z0,TT, sdX=1){
    Zt <- c(z0)

    Xt <- sampleMix(mean1=z0,mean2=(z0-1),mean3=(z0+1),sd=1)

    for (i in 2:TT){
        Zt[i] <- sampleMix(mean1=Zt[i-1],mean2=(Zt[i-1]+1),mean3=(Zt[i-1]+2),sdX=1)
        Xt[i] <- sampleMix(mean1=Zt[i],mean2=(Zt[i]-1),mean3=(Zt[i]+1),sdX=sdX)
    }
    return(list(sates=Zt,observations=Xt))

}

generateX <- function(Zt, TT, sdX=1){
  Xt <- c()
  for (i in 1:TT){
    Xt[i] <-  sampleMix(mean1=Zt[i],mean2=(Zt[i]-1),mean3=(Zt[i]+1),sdX=sdX)
  }
  return(Xt)
```

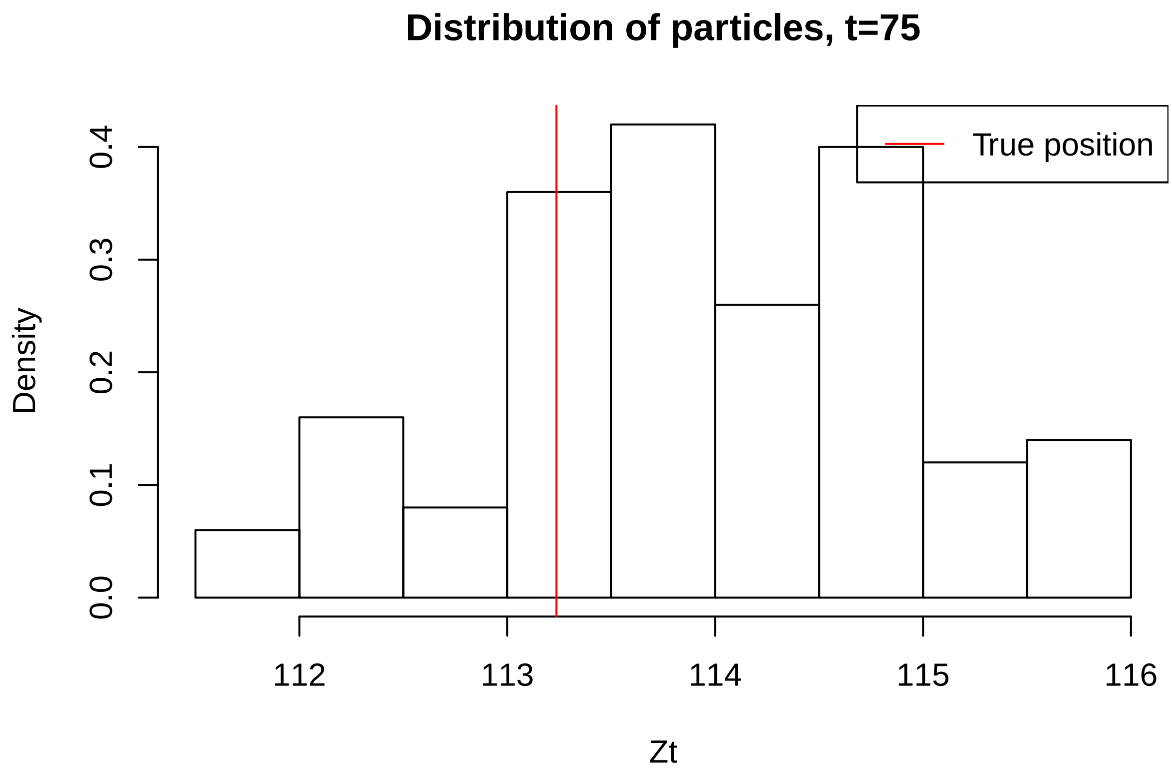
```r
}

## Particle filter function
particles <- function(Obs, M,Iter, sdX=1, correction=TRUE){
    Xt <- Obs

    #Initialise particles for the latent variable
    Zt <- matrix(ncol=Iter,nrow=M)
    wt <- matrix(ncol=Iter,nrow=M)
    Zt_bar <- matrix(ncol=Iter,nrow=M)
    #fill the initial time step
    Zt[,1] <- runif(M,0,100)
    # Add arbitrary values in order for the indexing to work later
    Zt_bar[,1] <- rep.int(1,M)
    wt[,1] <- rep.int(1,M)

    for (t in 2:Iter){
        # Prediction
        for (m in 1:M){
            # sample from the transition model (current Z_t depends on the Z_{t-1})
            Zt_bar[m,t] <- sampleMix(mean1=Zt[m,t-1],mean2=(Zt[m,t-1]+1),mean3=(Zt[m,t-1]+2),sdX=1)
            #Calculate weights
            wt[m,t] <- (dnorm(Xt[t],mean=Zt_bar[m,t],sd=sdX)+
                        dnorm(Xt[t],mean=(Zt_bar[m,t]-1),sd=sdX)+
                        dnorm(Xt[t],mean=(Zt_bar[m,t]+sdX),sd=1))/3
        }
        # Correction
        if (correction){
            Zt[,t]<- sample(Zt_bar[,t], M, replace = TRUE, prob = wt[,t])
        }else{
            Zt[,t] <- Zt_bar[,t]
        }

    }
    return(list(Zt=Zt,Zt_bar=Zt_bar,wt=wt))

}
```

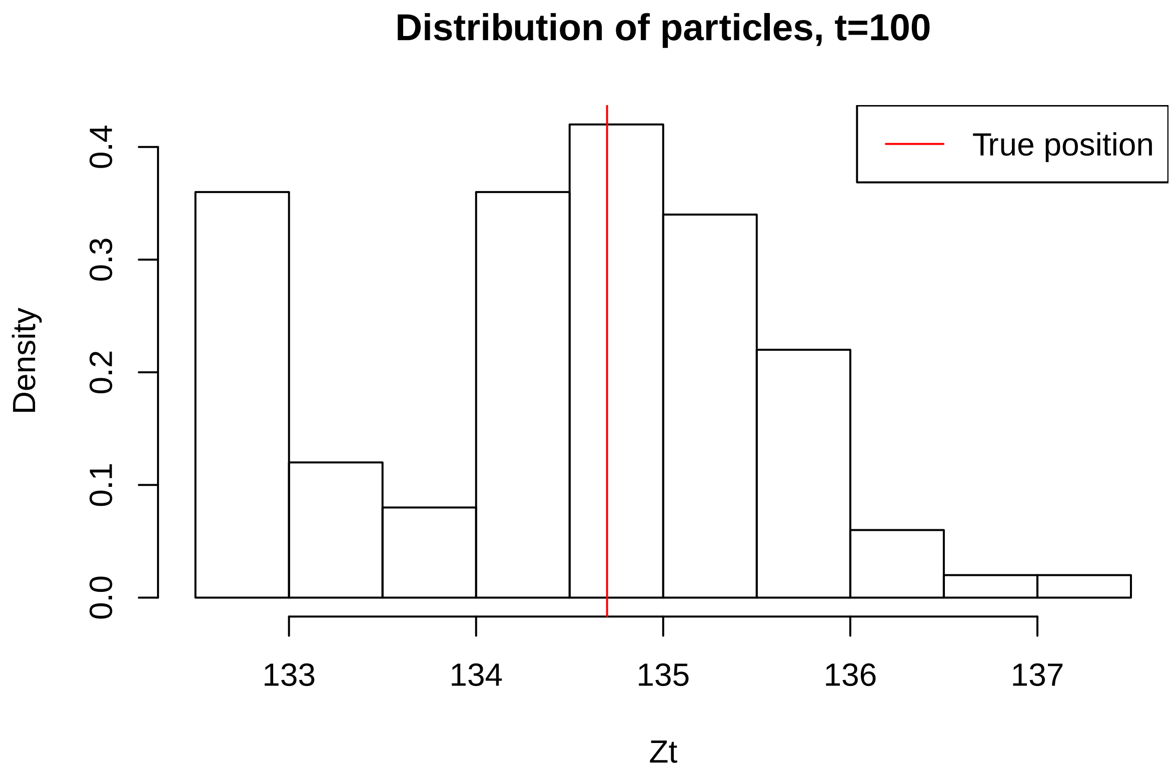
Afterward the states $z_{1:T}$ (using the initial and transition models) and observations $x_{1:T}$ of sensor readings (using the emission model) are simulated for $T = 100$ time steps. The observations are used in the particle filter algorithm with 100 particles in order to predict the possible states (location of the robot).

We chose time points 1, 35, 75, and 100 for which we plot the distribution of the particles (a histogram), and indicate on them what is the true value of the z_t at that particular time step.

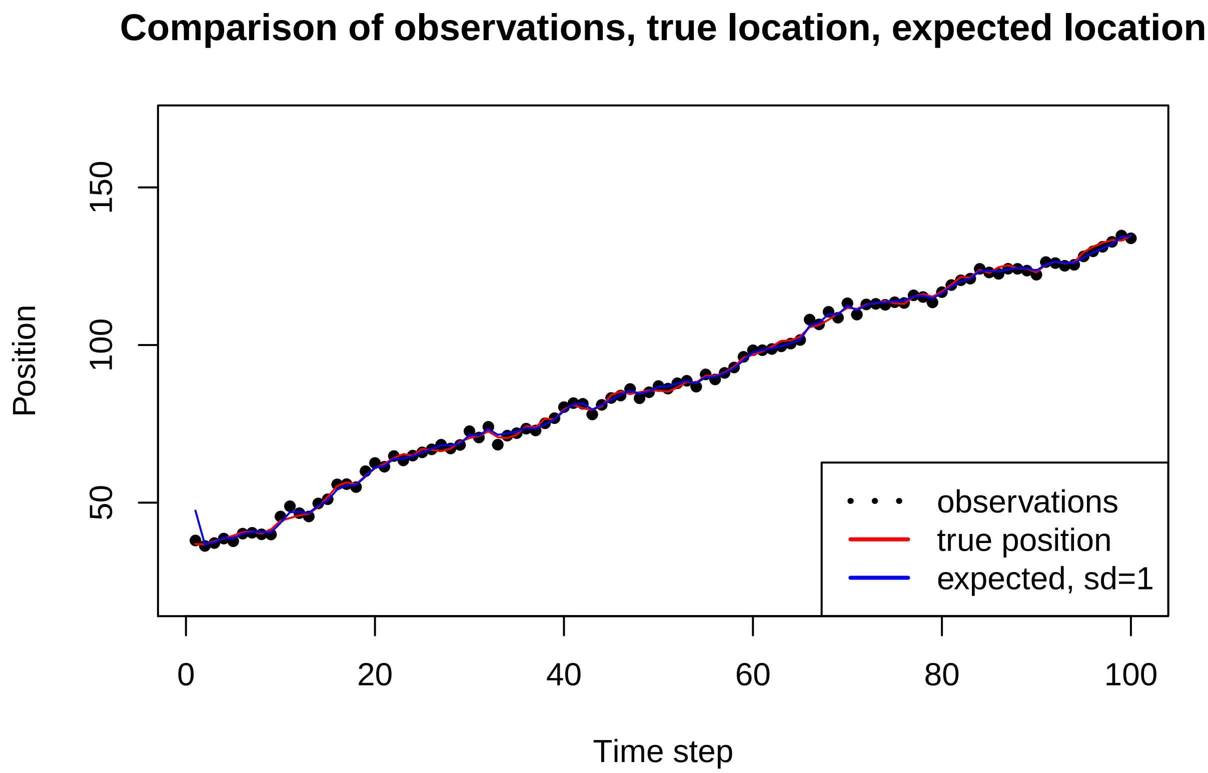

Distribution of particles, t=1
True position
Density
0.000
0.005
0.010
0.015
0
20
40
60
80
100
Zt


Distribution of particles, t=35
True position
Density
0.0
0.1
0.2
0.3
0.4
70
71
72
73
74
75
Zt


Distribution of particles, t=75
True position
Density
0.0
0.1
0.2
0.3
0.4
112
113
114
115
116
Zt



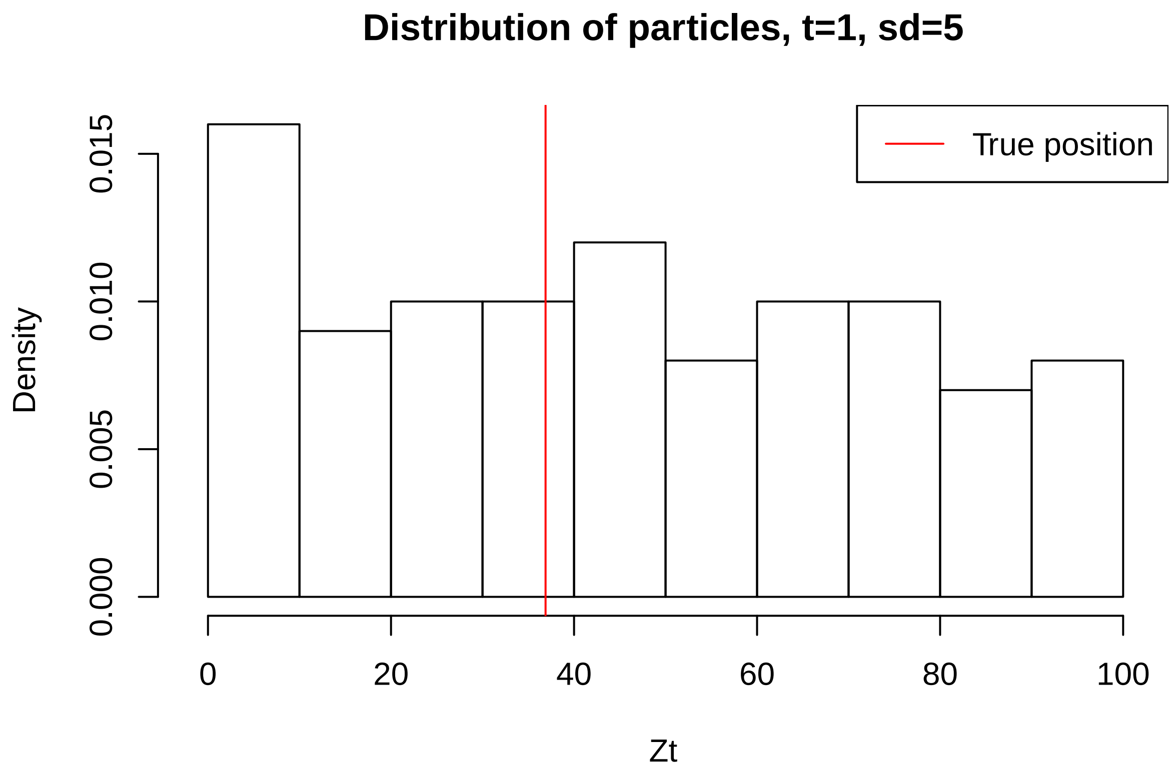
From the histograms, we conclude that the particle algorithm worked pretty well when the standard deviation was 1. In all four plots, the red line falls on one of the highest bars: this indicates that the algorithm quite accurately predicted the robots location (with a reasonable error).

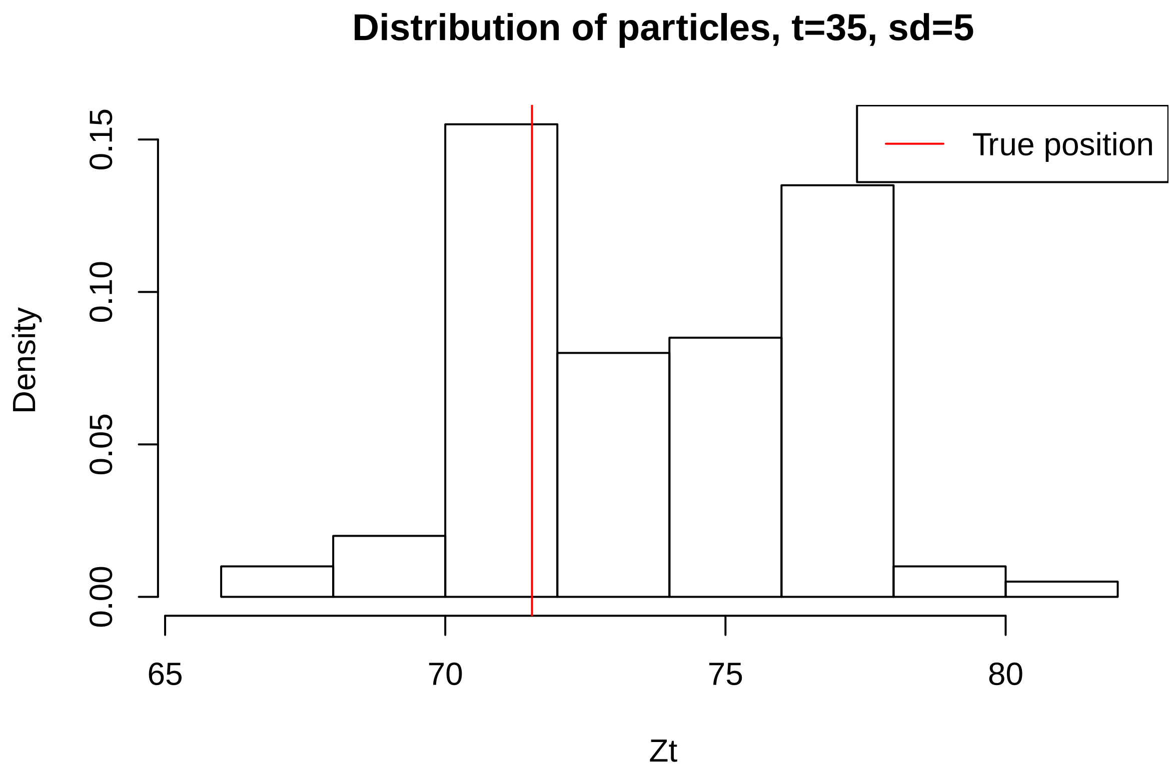


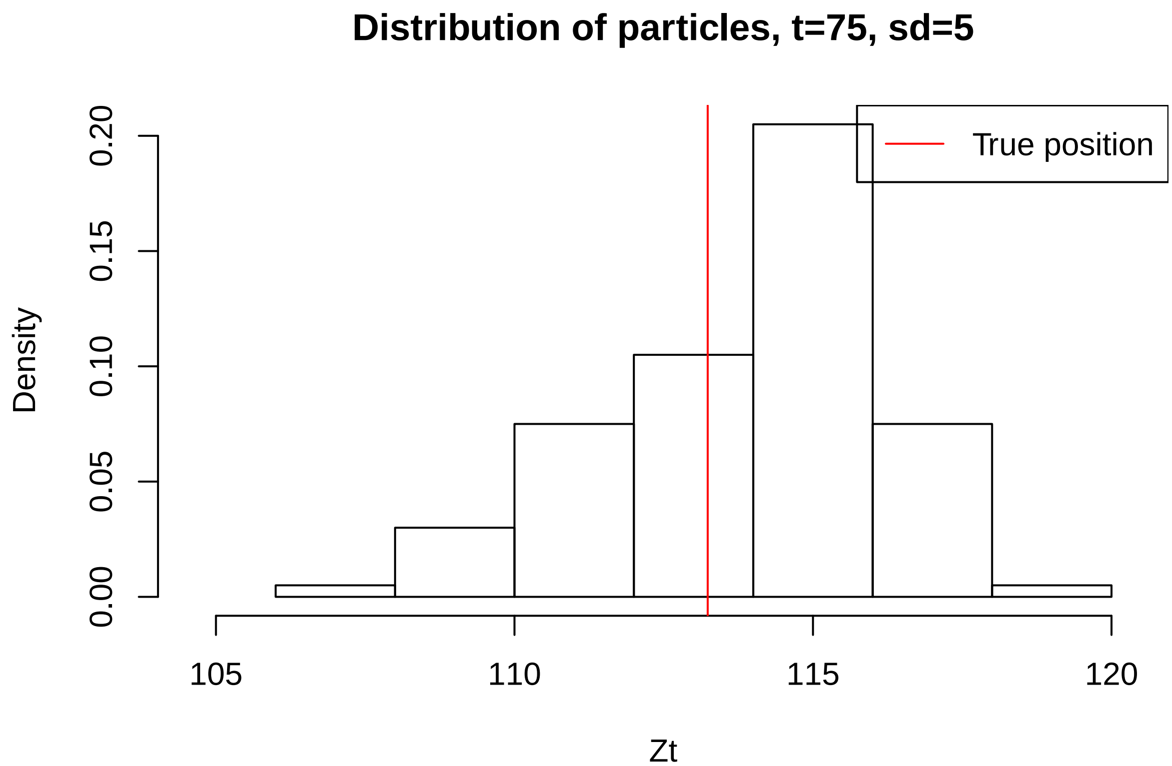
The plot above shows the averge of particles in each time step, the corresponding actual state and observations. We can see that the true position and predicted positions are very close to each other, so the prediction seems to be very accurate.

Question 2

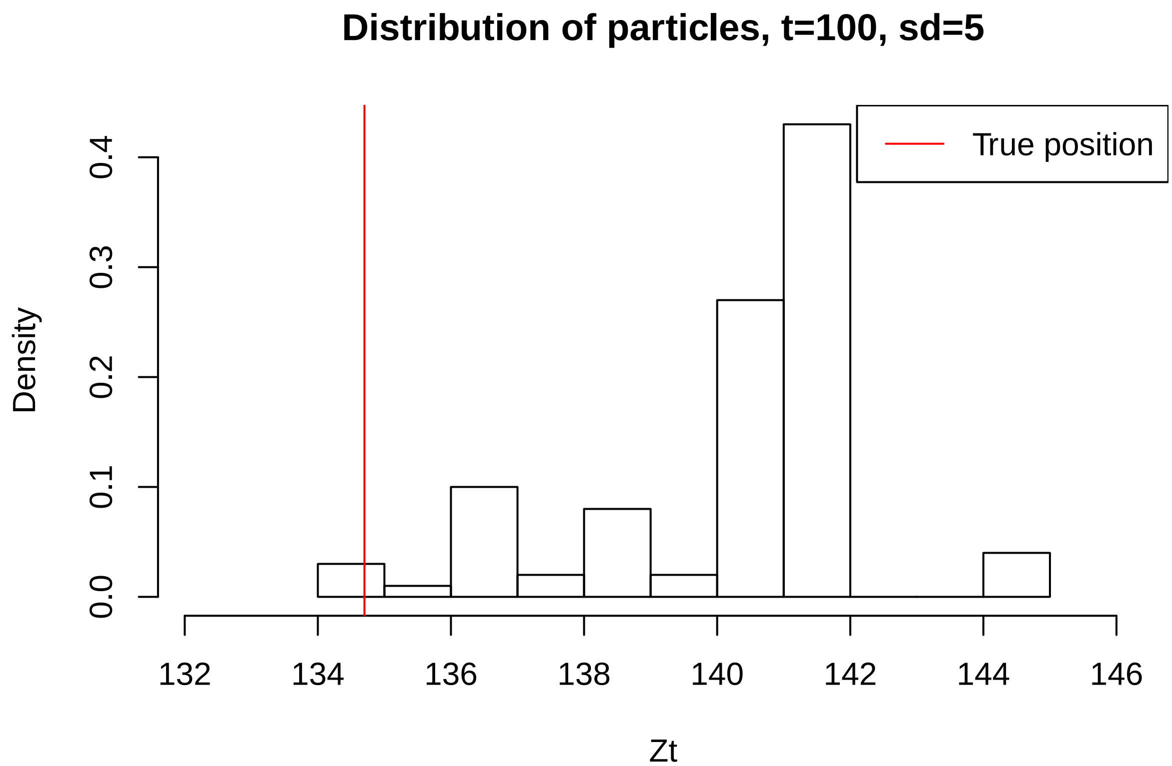
In this question the HMM is modified by changing the emission models. Firstly, the standard deviation is set to 5, and later it is increased to 50. The states $z_{1:T}$ are kept the same, but new observations $x_{1:T}$ are generated. The particle filter algorithm is applied again on both cases.


Distribution of particles, t=1, sd=5
True position
Density
0.000
0.005
0.010
0.015
0
20
40
60
80
100
Zt

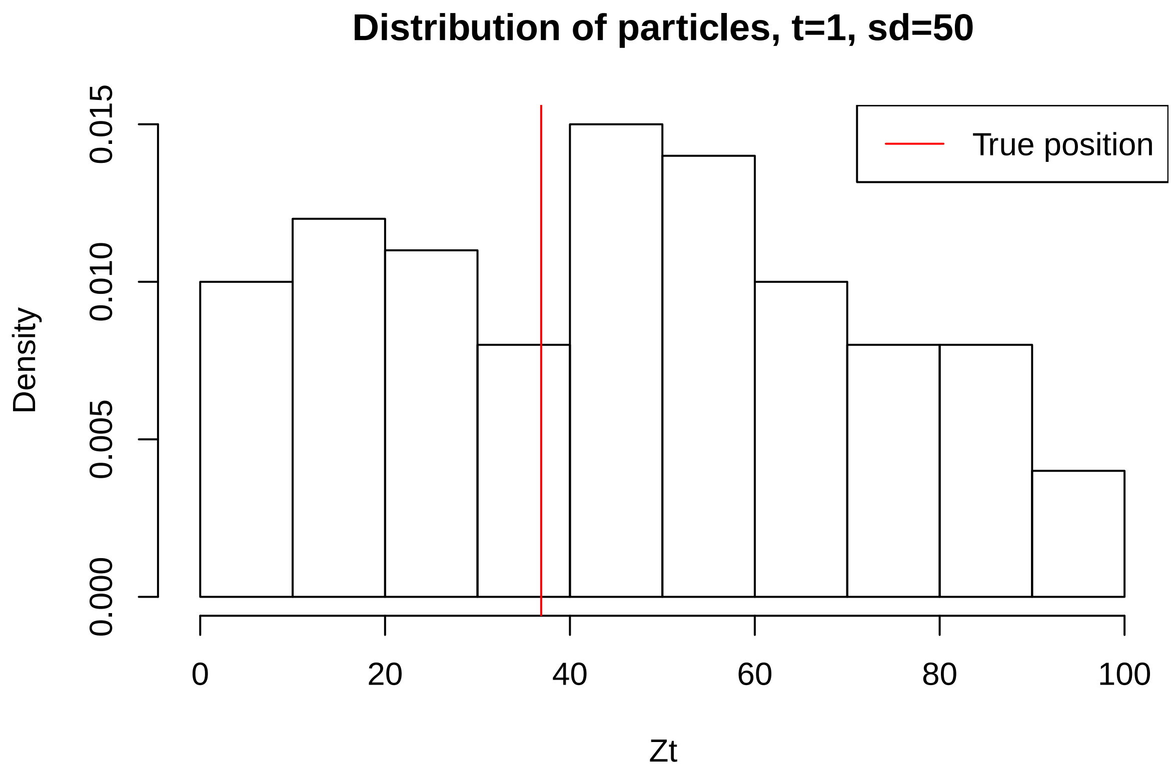

Distribution of particles, t=35, sd=5
True position
Density
0.00
0.05
0.10
0.15
65
70
75
80
Zt

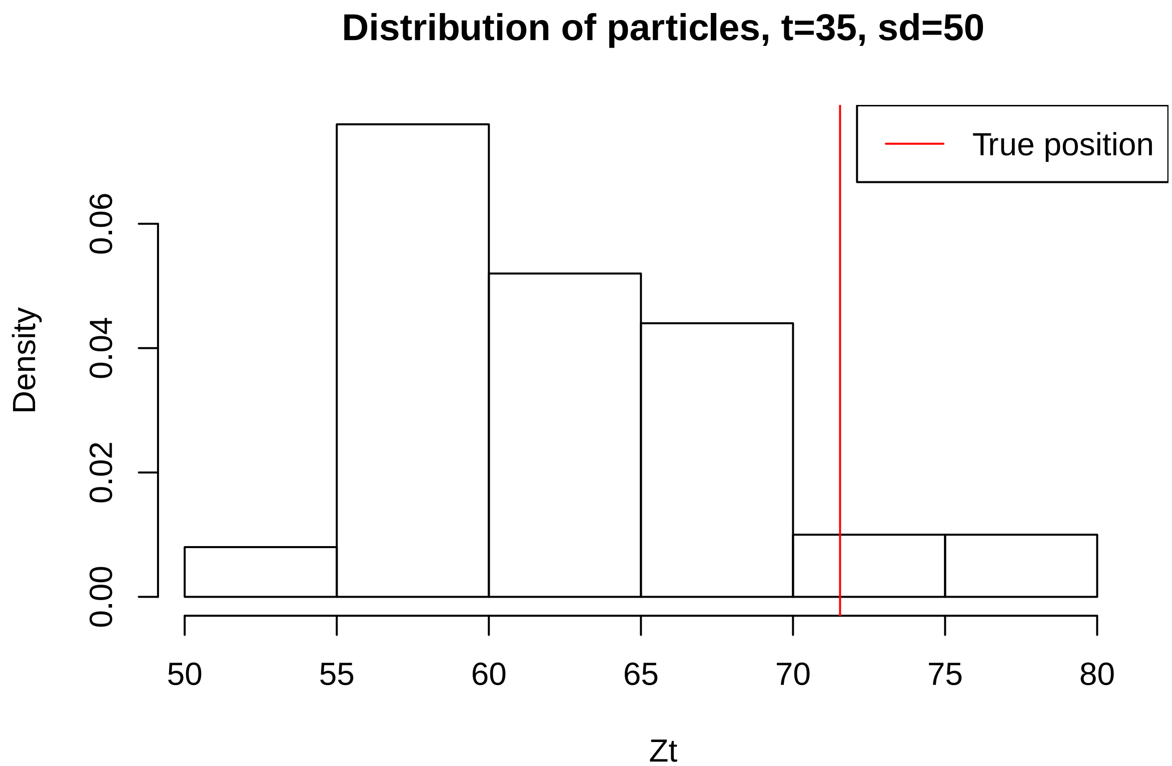


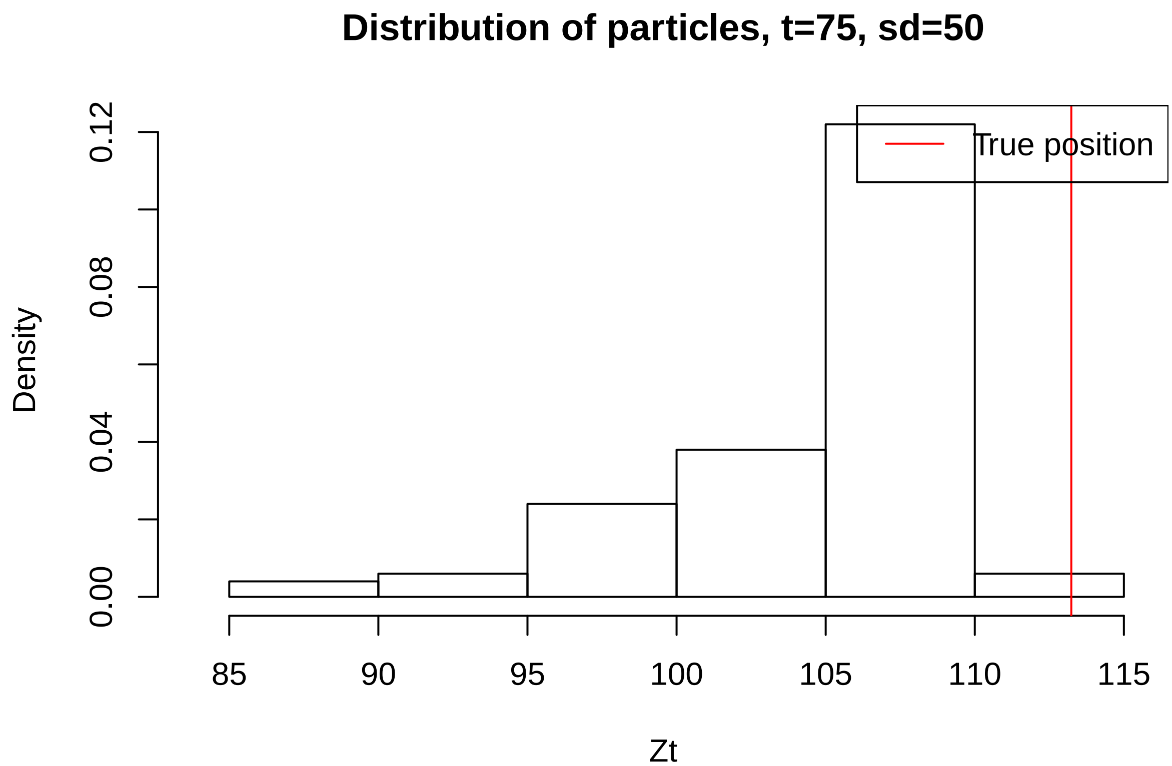
Distribution of particles, t=75, sd=5
True position
Density
0.00
0.05
0.10
0.15
0.20
105
110
115
120
Zt



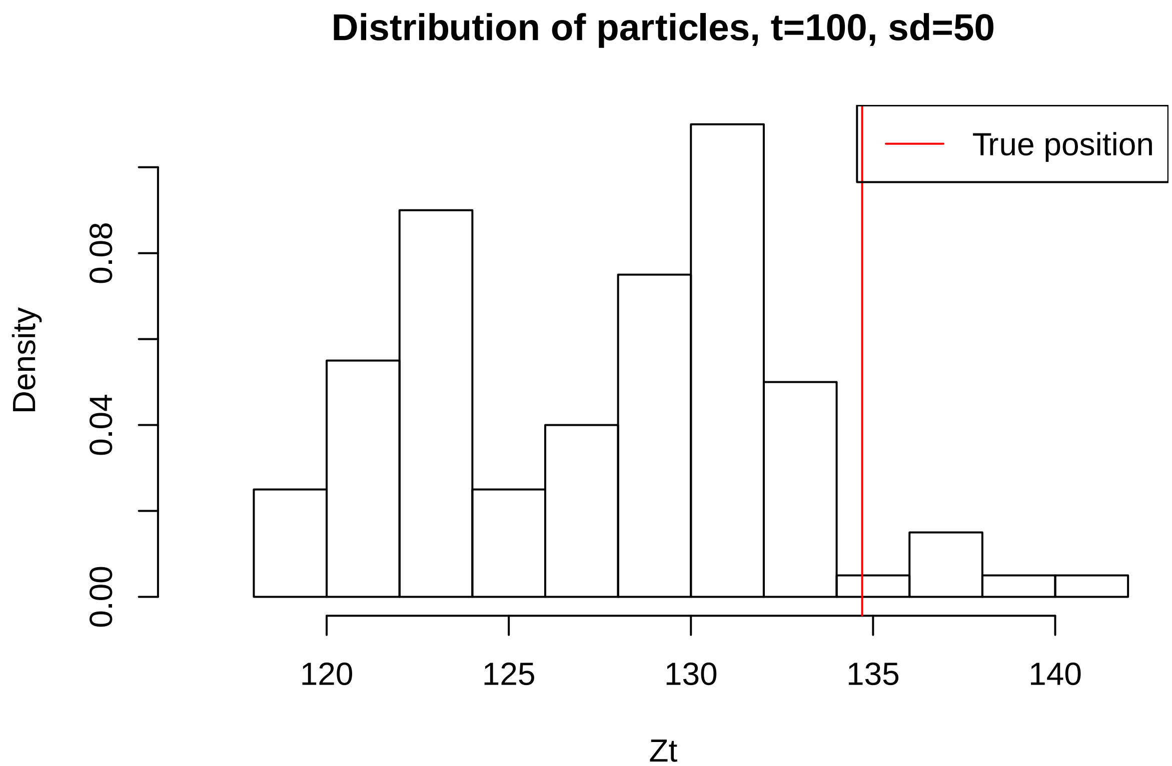
The histogram plots above show the distribution of the particles when standard deviation is 5 (for time steps 1, 35, 75, and 100). This time the algorithm possibly performed worse: at time step 100 the true location of the robot falls into one of the shortest bars of the distribution. This indicates that the algorithm predicted quite a low probability that z_100 is the true value. However, the overall performance cannot be deduced by looking at only 4 time points: each new step introduces some new uncertainty, and the results at it might be worse then the results from the previous time step.


Distribution of particles, t=1, sd=50
True position
Density
0.000
0.005
0.010
0.015
0
20
40
60
80
100
Zt

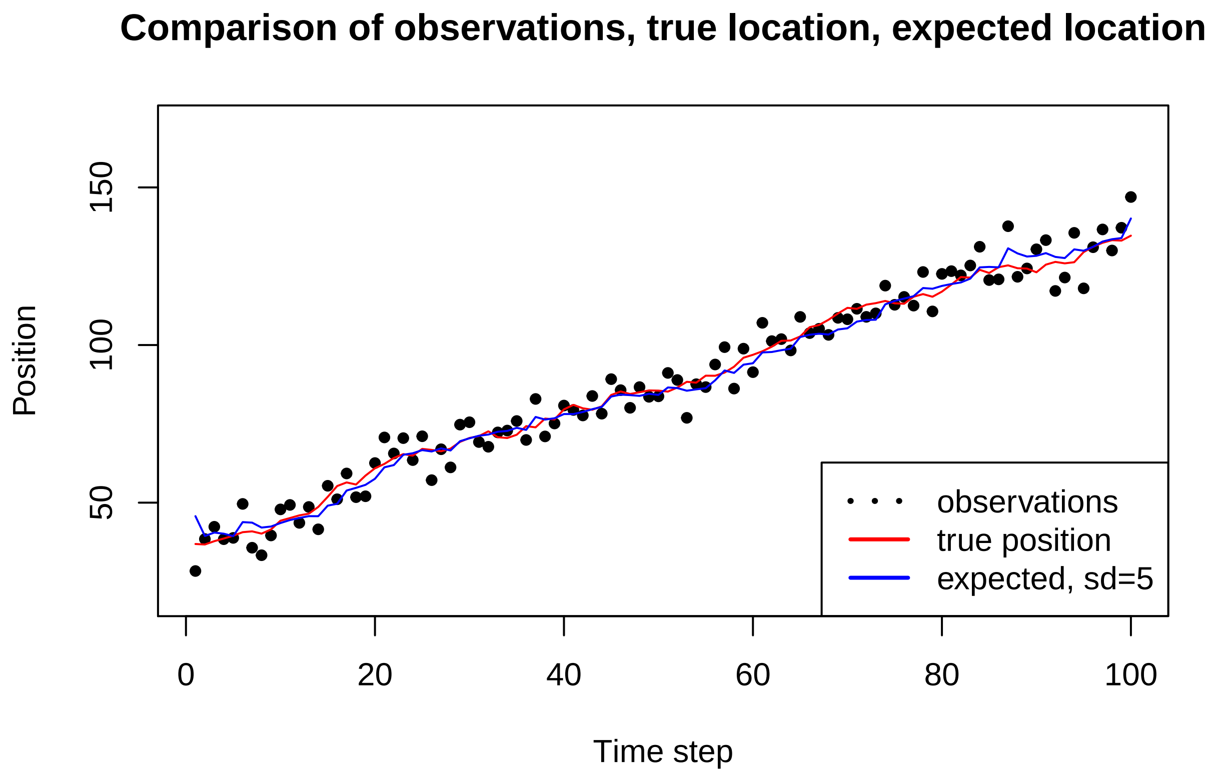

Distribution of particles, t=35, sd=50
True position
Density
0.00
0.02
0.04
0.06
50
55
60
65
70
75
80
Zt

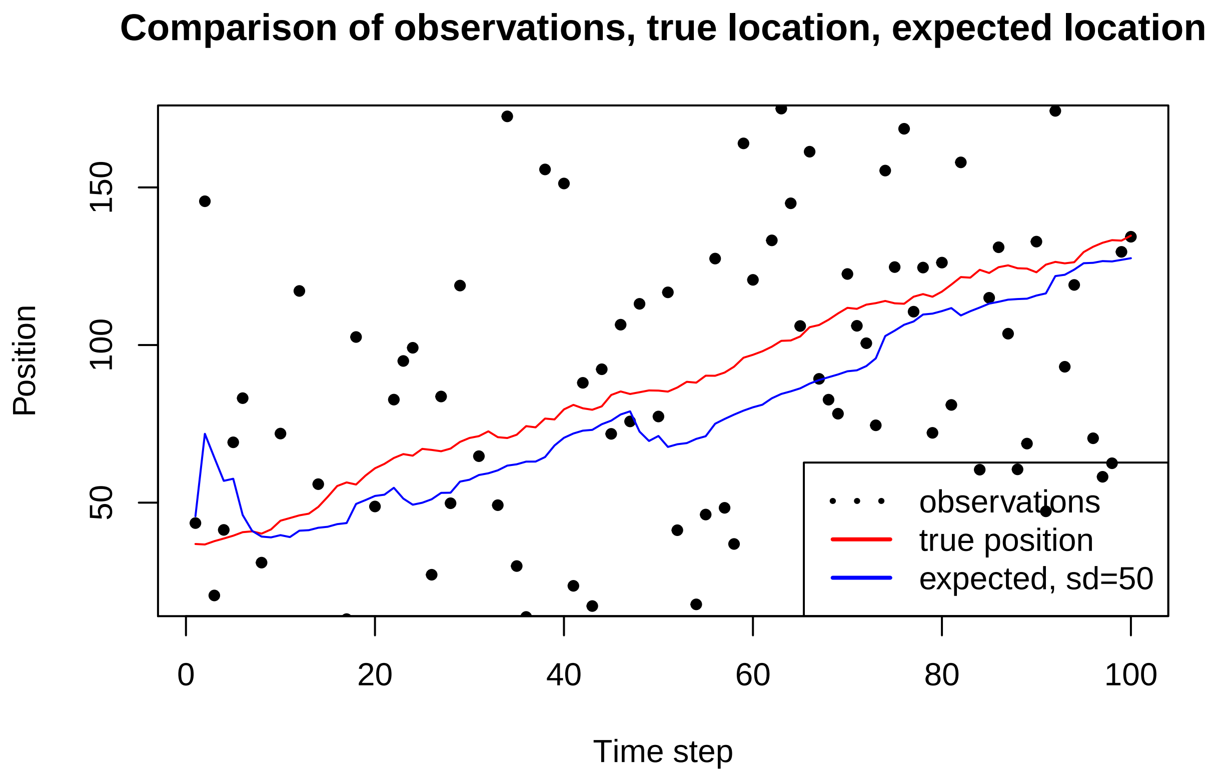


Distribution of particles, t=75, sd=50
True position
Density
0.00
0.04
0.08
0.12
85
90
95
100
105
110
115
Zt



The histogram plots above show the distribution of the particles when standard deviation is 50 (for time steps 1, 35, 75, and 100). This time the true location of the robot for time steps 35, 75, and 100 falls into one of the shortest bars of the distribution of that time step. This indicates that the algorithm predicted quite a low probability that z_t is the true value. It is reasonable to assume that the performance of the particle filter worsens as the standard deviation of the observations from the true states is increased: there is more uncertainty involved.

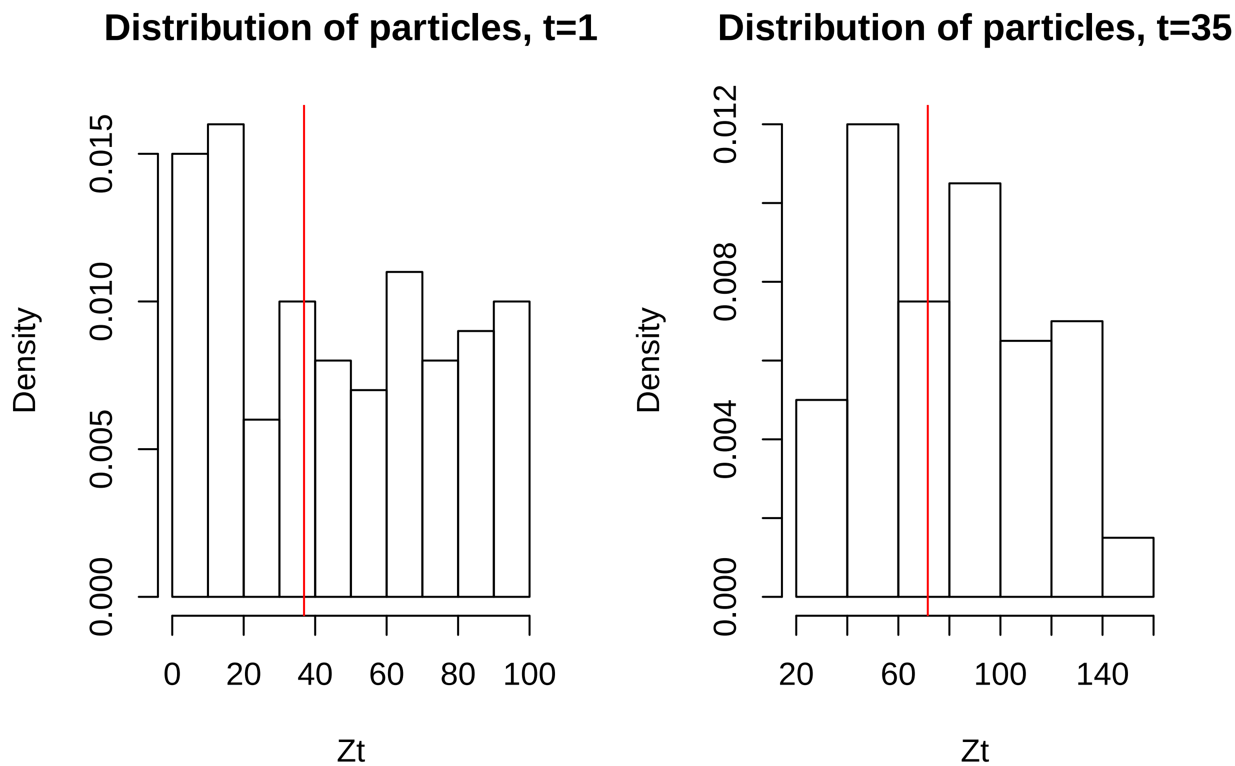

Comparison of observations, true location, expected location
Position
150
100
50
0
20
40
60
80
100
Time step
observations
true position
expected, sd=5

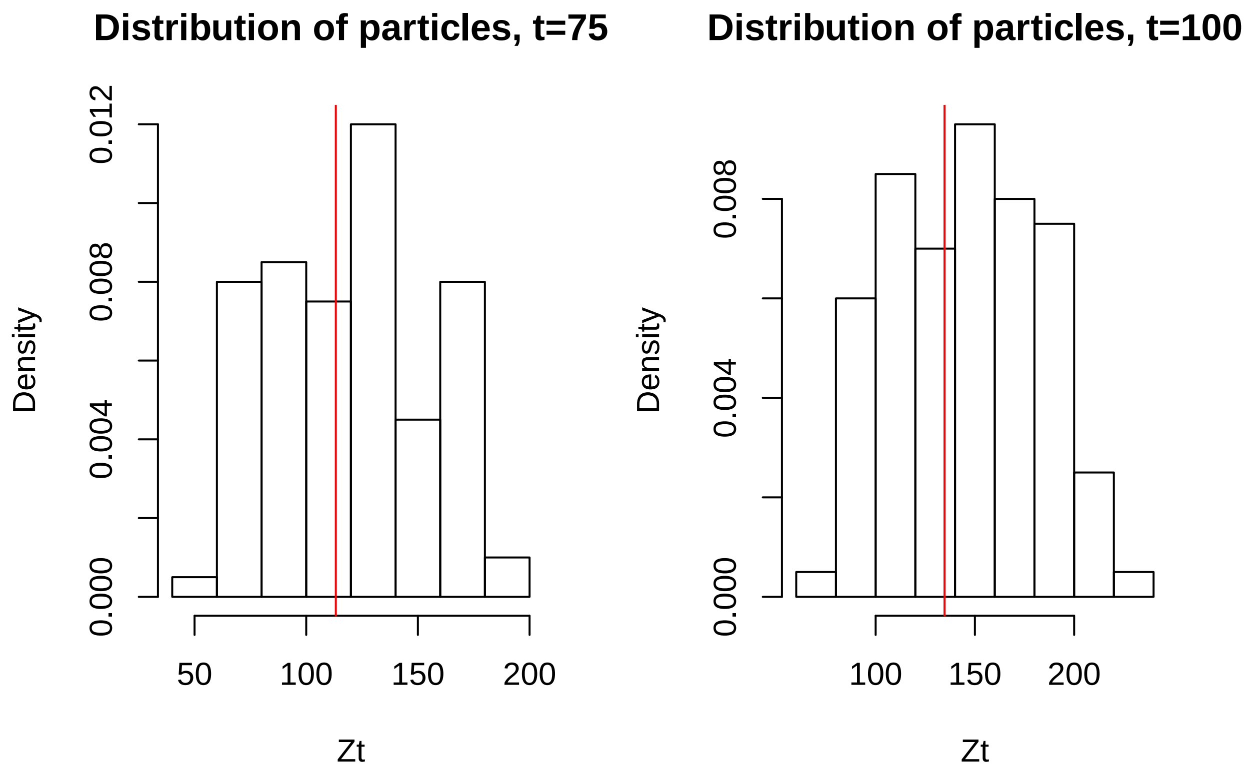


From the plots above, the observations and expected location deviate more from the true position when the standard deviation is 50 than they did when standard devation was 5. This is because $sd = 50$ adds even more uncertainty to the observations and thus the expected location than $sd = 5$.

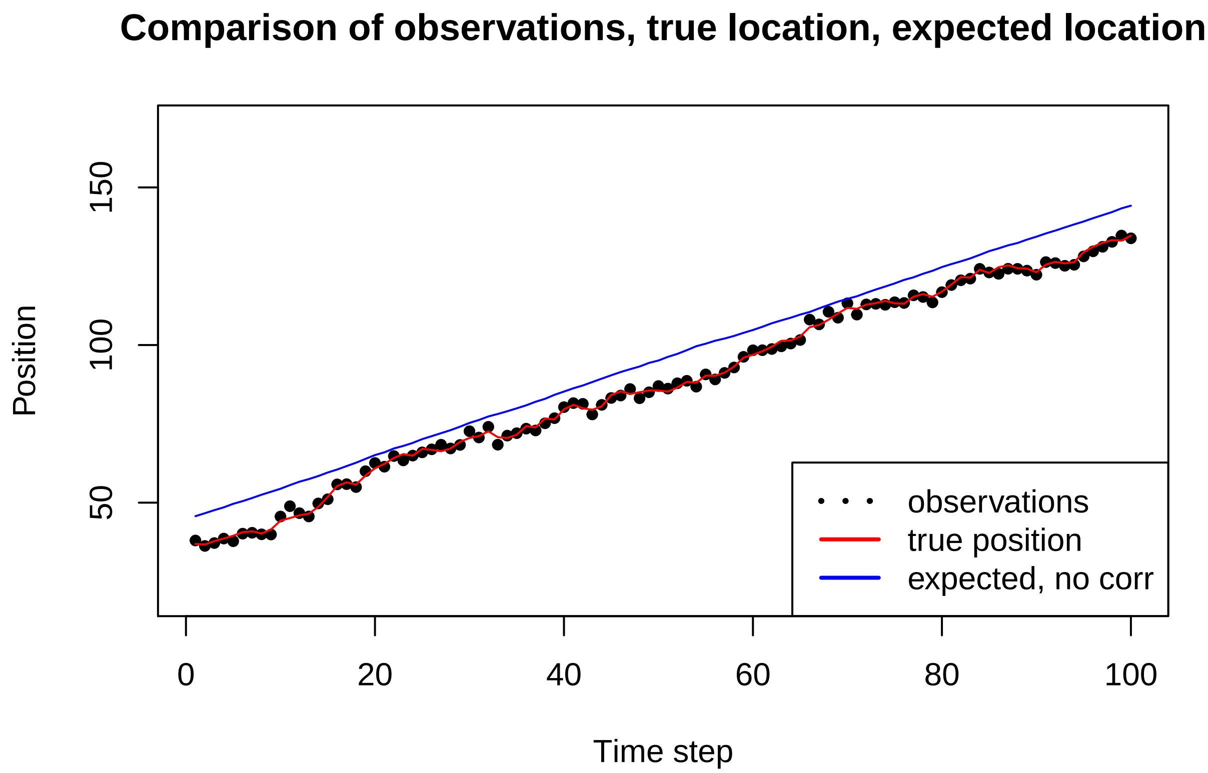
Question 3

Question 1 was repeated without correction, i.e. importance weights were always equal to 1.


Distribution of particles, t=1
Density
0.000
0.005
0.010
0.015
0
20
40
60
80
100
Zt
Distribution of particles, t=35
Density
0.000
0.004
0.008
0.012
20
60
100
140
Zt



From the distribution plots, it seems that the particle filtering algorithm without correction step did not performed much worse than for the same model with correction.



Without correction, the expected location follows a straight line. It resembles a linear regression. Without correction, unlikely particles continues to the next time step. With correction, unlikely particles are less likely to be selected to continue to the next time step and therefore the expected location will deviate less from the true position.

Appendix

```
knitr::opts_chunk$set(echo = FALSE,tidy.opts=list(width.cutoff=60),tidy=TRUE)
set.seed(987654321)
# M - number of particals
# Iter - number of time steps T
# z is latent variable
# x is the observation

# Function for sampling from a mixture model
sampleMix <- function(mean1,mean2,mean3,sdX){
   u <- sample(1:3,1)
   means <- c(mean1,mean2,mean3)
   return(rnorm(1,mean=means[u],sd=sdX))
}

## Function that creates the SSM model
create_SSM <- function(z0,TT, sdX=1){
    Zt <- c(z0)

    Xt <- sampleMix(mean1=z0,mean2=(z0-1),mean3=(z0+1),sd=1)

    for (i in 2:TT){
        Zt[i] <- sampleMix(mean1=Zt[i-1],mean2=(Zt[i-1]+1),mean3=(Zt[i-1]+2),sdX=1)
        Xt[i] <- sampleMix(mean1=Zt[i],mean2=(Zt[i]-1),mean3=(Zt[i]+1),sdX=sdX)
    }
    return(list(sates=Zt,observations=Xt))

}

generateX <- function(Zt, TT, sdX=1){
  Xt <- c()
  for (i in 1:TT){
    Xt[i] <-  sampleMix(mean1=Zt[i],mean2=(Zt[i]-1),mean3=(Zt[i]+1),sdX=sdX)
  }
  return(Xt)
}

## Particle filter function
particles <- function(Obs, M,Iter, sdX=1, correction=TRUE){
    Xt <- Obs

    #Initialise particles for the latent variable
    Zt <- matrix(ncol=Iter,nrow=M)
    wt <- matrix(ncol=Iter,nrow=M)
    Zt_bar <- matrix(ncol=Iter,nrow=M)
    #fill the initial time step
    Zt[,1] <- runif(M,0,100)
    # Add arbitrary values in order for the indexing to work later
    Zt_bar[,1] <- rep.int(1,M)
    wt[,1] <- rep.int(1,M)

    for (t in 2:Iter){
```

```r
        # Prediction
        for (m in 1:M){
            # sample from the transition model (current Z_t depends on the Z_{t-1})
            Zt_bar[m,t] <- sampleMix(mean1=Zt[m,t-1],mean2=(Zt[m,t-1]+1),mean3=(Zt[m,t-1]+2),sdX=1)
            #Calculate weights
            wt[m,t] <- (dnorm(Xt[t],mean=Zt_bar[m,t],sd=sdX)+
                        dnorm(Xt[t],mean=(Zt_bar[m,t]-1),sd=sdX)+
                        dnorm(Xt[t],mean=(Zt_bar[m,t]+sdX),sd=1))/3
        }
        # Correction
        if (correction){
            Zt[,t]<- sample(Zt_bar[,t], M, replace = TRUE, prob = wt[,t])
        }else{
            Zt[,t] <- Zt_bar[,t]
        }

    }
    return(list(Zt=Zt,Zt_bar=Zt_bar,wt=wt))

}

# Create the model
my_SSM <- create_SSM(runif(1,0,100),100)

# Filter particles
results_q1 <- particles(my_SSM$observations,100,100)
hist(results_q1$Zt[,1],freq = FALSE,xlab = "Zt",main="Distribution of particles, t=1")
abline(v=my_SSM$sates[1],col="red")
legend("topright",c("True position"),
       col = c("red"),lty = c(1))

hist(results_q1$Zt[,35],freq = FALSE,xlab = "Zt",main="Distribution of particles, t=35")
abline(v=my_SSM$sates[35],col="red")
legend("topright",c("True position"),
       col = c("red"),lty = c(1))

hist(results_q1$Zt[,75],freq = FALSE,xlab = "Zt",main="Distribution of particles, t=75")
abline(v=my_SSM$sates[75],col="red")
legend("topright",c("True position"),
       col = c("red"),lty = c(1))

hist(results_q1$Zt[,100],freq = FALSE,xlab = "Zt",main="Distribution of particles, t=100")
abline(v=my_SSM$sates[100],col="red")
legend("topright",c("True position"),
       col = c("red"),lty = c(1))
exp_loc_q1 <- colMeans(results_q1$Zt)
plot(my_SSM$observations, pch=20, ylab="Position", xlab="Time step",
     main="Comparison of observations, true location, expected location", ylim = c(20,170))
lines(my_SSM$sates, col="red")
lines(exp_loc_q1, col="blue")
legend("bottomright",c("observations","true position","expected, sd=1"),
       col = c("black","red","blue"), lty = c(3,1, 1), lwd=c(3,rep(2,2)))
```

```r
# Create observations for the model with sd=5
my_SSM_2a <-list()
my_SSM_2a$sates <- my_SSM$sates
my_SSM_2a$observations <- generateX(my_SSM$sates,100,sdX = 5)
# Filter particles, sd=5
results_q2a <- particles(my_SSM_2a$observations,100,100,sdX = 5)

# Create the model, sd=50
my_SSM_2b <-list()
my_SSM_2b$sates <- my_SSM$sates
my_SSM_2b$observations <- generateX(my_SSM$sates,100,sdX = 50)
# Filter particles, sd=50
results_q2b <- particles(my_SSM_2b$observations,100,100,sdX = 50)

# Plot results from 2a, sd =5
hist(results_q2a$Zt[,1],freq = FALSE,xlab = "Zt",main="Distribution of particles, t=1, sd=5",
     xlim=c(min(min(results_q2a$Zt[,1],my_SSM_2a$sates[1]))-2,
            max(max(results_q2a$Zt[,1]),my_SSM_2a$sates[1])+2))
abline(v=my_SSM_2a$sates[1],col="red")
legend("topright",c("True position"),
       col = c("red"),lty = c(1))

hist(results_q2a$Zt[,35],freq = FALSE,xlab = "Zt",main="Distribution of particles, t=35, sd=5",
     xlim=c(min(min(results_q2a$Zt[,35],my_SSM_2a$sates[35]))-2,
            max(max(results_q2a$Zt[,35]),my_SSM_2a$sates[35])+2))
abline(v=my_SSM_2a$sates[35],col="red")
legend("topright",c("True position"),
       col = c("red"),lty = c(1))

hist(results_q2a$Zt[,75],freq = FALSE,xlab = "Zt",main="Distribution of particles, t=75, sd=5",
     xlim=c(min(min(results_q2a$Zt[,75],my_SSM_2a$sates[75]))-2,
            max(max(results_q2a$Zt[,75]),my_SSM_2a$sates[75])+2))
abline(v=my_SSM_2a$sates[75],col="red")
legend("topright",c("True position"),
       col = c("red"),lty = c(1))

hist(results_q2a$Zt[,100],freq = FALSE,xlab = "Zt",main="Distribution of particles, t=100, sd=5",
     xlim=c(min(min(results_q2a$Zt[,100],my_SSM_2a$sates[100]))-2,
            max(max(results_q2a$Zt[,100]),my_SSM_2a$sates[100])+2))
abline(v=my_SSM_2a$sates[100],col="red")
legend("topright",c("True position"),
       col = c("red"),lty = c(1))


# Plot results from 2b, sd =50
hist(results_q2b$Zt[,1],freq = FALSE,xlab = "Zt",main="Distribution of particles, t=1, sd=50",xlim=c(mi
              max(max(results_q2b$Zt[,1]),my_SSM_2b$sates[1])+2))
abline(v=my_SSM_2b$sates[1],col="red")
legend("topright",c("True position"),
       col = c("red"),lty = c(1))


hist(results_q2b$Zt[,35],freq = FALSE,xlab = "Zt",main="Distribution of particles, t=35, sd=50",
```

```
     xlim=c(min(min(results_q2b$Zt[,35],my_SSM_2b$sates[35]))-2,
            max(max(results_q2b$Zt[,35]),my_SSM_2b$sates[35])+2))
abline(v=my_SSM_2b$sates[35],col="red")
legend("topright",c("True position"),
       col = c("red"),lty = c(1))


hist(results_q2b$Zt[,75],freq = FALSE,xlab = "Zt",main="Distribution of particles, t=75, sd=50",
     xlim=c(min(min(results_q2b$Zt[,75],my_SSM_2b$sates[75]))-2,
            max(max(results_q2b$Zt[,75]),my_SSM_2b$sates[75])+2))
abline(v=my_SSM_2b$sates[75],col="red")
legend("topright",c("True position"),
       col = c("red"),lty = c(1))


hist(results_q2b$Zt[,100],freq = FALSE,xlab = "Zt",main="Distribution of particles, t=100, sd=50",
     xlim=c(min(min(results_q2b$Zt[,100],my_SSM_2b$sates[100]))-2,
            max(max(results_q2b$Zt[,100]),my_SSM_2b$sates[100])+2))
abline(v=my_SSM_2b$sates[100],col="red")
legend("topright",c("True position"),
       col = c("red"),lty = c(1))

exp_loc_q2a <- colMeans(results_q2a$Zt)
exp_loc_q2b <- colMeans(results_q2b$Zt)



plot(my_SSM_2a$observations, pch=20, ylab="Position", xlab="Time step",
     main="Comparison of observations, true location, expected location", ylim = c(20,170))
lines(my_SSM$sates, col="red")
lines(exp_loc_q2a, col="blue")
legend("bottomright",c("observations","true position","expected, sd=5"),
       col = c("black","red","blue"), lty = c(3,1, 1), lwd=c(3,rep(2,2)))

plot(my_SSM_2b$observations, pch=20, ylab="Position", xlab="Time step",
     main="Comparison of observations, true location, expected location", ylim = c(20,170))
lines(my_SSM$sates, col="red")
lines(exp_loc_q2b, col="blue")
legend("bottomright",c("observations","true position","expected, sd=50"),
       col = c("black","red","blue"), lty = c(3,1, 1), lwd=c(3,rep(2,2)))
# Filter particles
results_q3 <- particles(my_SSM$observations,100,100,correction = FALSE)
par(mfrow=c(1,2))

hist(results_q3$Zt[,1],freq = FALSE,xlab = "Zt",main="Distribution of particles, t=1")
abline(v=my_SSM$sates[1],col="red")


hist(results_q3$Zt[,35],freq = FALSE,xlab = "Zt",main="Distribution of particles, t=35")
abline(v=my_SSM$sates[35],col="red")


hist(results_q3$Zt[,75],freq = FALSE,xlab = "Zt",main="Distribution of particles, t=75")
```

```r
abline(v=my_SSM$sates[75],col="red")


hist(results_q3$Zt[,100],freq = FALSE,xlab = "Zt",main="Distribution of particles, t=100")
abline(v=my_SSM$sates[100],col="red")


exp_loc_q3 <- colMeans(results_q3$Zt)



plot(my_SSM$observations, pch=20, ylab="Position", xlab="Time step",
     main="Comparison of observations, true location, expected location", ylim = c(20,170))
lines(my_SSM$sates, col="red")
lines(exp_loc_q3, col="blue")
legend("bottomright",c("observations","true position","expected, no corr"),
       col = c("black","red","blue"), lty = c(3,1, 1), lwd=c(3,rep(2,2)))
```