ON THE FLY VISION SYSTEM FOR AIR BEARING SURFACE TRACKING

THANEE SAMSICHAROENLAP

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF

MASTER OF ENGINEERING IN DATA STORAGE TECHNOLOGY

INTERNATIONAL COLLEGE

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2014

KMITL-2014-IC-M-005-007

ON THE FLY VISION SYSTEM FOR AIR BEARING SURFACE TRACKING

THANEE SAMSICHAROENLAP

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENT FOR THE DEGREE OF
MASTER OF ENGINEERING IN DATA STORAGE TECHNOLOGY
INTERNATIONAL COLLEGE
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG
2014

KMITL-2014-IC-M-005-007

Thesis Title: On The Fly Vision System For Air Bearing Surface Tracking

Student: Thanee Samsicharoenlap

Student ID: 54600702

Degree: Master of Engineering

Program: Data Storage Technology

Year: 2014

Thesis Advisor: Dr.Somyot Kaitwanidvilai

ABSTRACT

Nowadays, the improvement in manufacturing process in the hard disk drive industry is required to achieve nanotechnology. Vision system has been extensively used in machine inspection and measurement. However, the visual inspection carried out by human consumes time and results in poor accuracy. Thus, in this paper, we propose of feasibility study in automatic visual system for tracking the air bearing surface (ABS) on a slider wafer. In addition, the non-stop vision system called “vision on the fly” is developed to enhance the productivity in terms of unit per hour (UPH). Particle filter is adopted to determine the position of the interesting object on the captured video even the object is moving. Based on the results, there is feasibility of UPH can be improved more than 20% by the proposed technique.

ACKNOWLEDGEMENT

This thesis would not have been possible without the guidance and the support of several persons who contributed and extended their appreciated assistance in the completion of this research.

This research work is supported by King Mongkut's Institute of Technology Research Fund. This work is also supported by the Seagate (Thailand) Co. Ltd. and DSTAR, KMITL

Thanee Samsicharoenlap

CONTENTS

CONTENTS (CONT.)

LIST OF TABLES

Tables **Page**

LIST OF FIGURES

LIST OF FIGURES (CONT.)

LIST OF FIGURES (CONT.)

LIST OF FIGURES (CONT.)

Figures **Page**

CHAPTER 1
INTRODUCTION

1.1 Background and Problems

Vision system has been widely utilized in the manufacturing process for many kinds of applications such as positioning, inspection, vision system has been continuously improved by many algorithms to achieve higher accuracy and speed of the inspection. However, for extremely precise application, the object of interest must be stationary or stop at the inspection point. For example, the inspection of slider wafer needs about 1.5 second for stopping the pallet before performing the visual inspection. The machine vision system must wait until slider wafer completely stop before capturing the image, spending more time caused by motor stopping and starting. To increase the inspection speed the unit per hour (UPH) of the machine is increased.

The inspection of slider wafer need about 1.5 seconds for stopping the robot before performing the visual inspection.

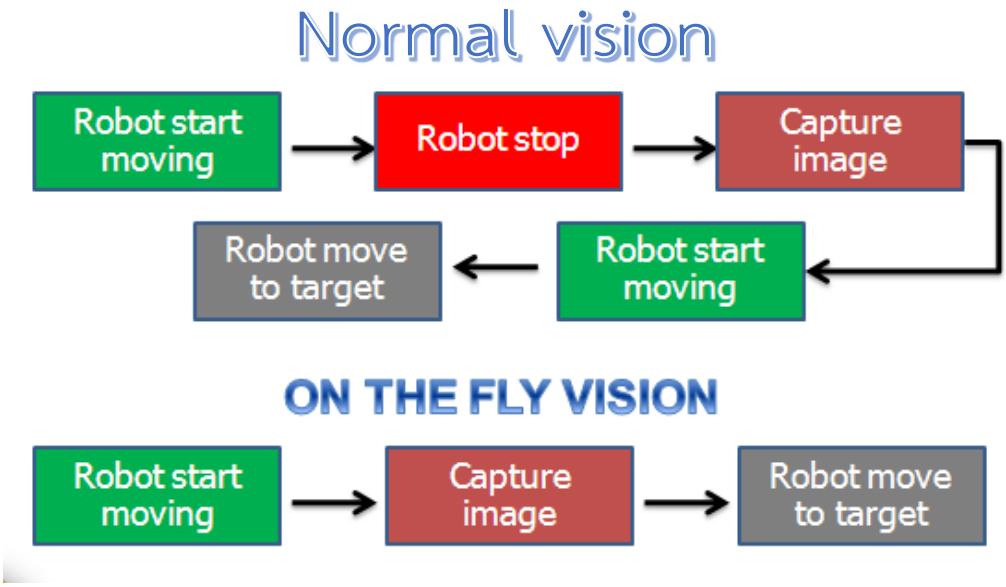


Figure 1.1 Machine flow chart

1.2 Objective

In order to increase unit per hour for reducing cost of the factory, this research focuses on the feasibility of detect moving slider wafer by using particle filter and resampling according to likelihood function technique to come over the robot is stopped while inspection slider wafer issue.

1.3 Scopes

1.3.1 Design a slider wafer detection using particle filter and resampling according to likelihood function on MATLAB for detect slider on the vision screen.

1.3.2 To study the parameter of the function from detecting result of slider wafer movement

1.4 Expected Benefits

1.4.1 Being able to solve the issue of current robot stopped for inspecting system by proposed system of particle filter using resampling according to likelihood.

1.4.2 Increase the unit per hour about 20%.

CHAPTER 2
LITERATURE REVIEW

The particle filter algorithm is well known as a very effective solution for handling nonlinear problems, many time series problems arising in statistics, engineering and applied sciences are concerned with the estimation of the state of a dynamic model when only inaccurate observations are available. Most real-world problems are nonlinear and non-Gaussian, There are many variations of particle filter Thus, there are many application use particle filter for detecting object [1-7]

2.1 Flood Water Level Prediction and Tracking using Particle Filter Algorithm

This algorithm is applied to predict the flood water level [1], flood water level prediction and monitoring as an alarming system to prevent flood disasters. This paper proposes Sequential Importance Sampling (SIS) particle filter to solve the above mentioned problem compare with Sampling Importance Resampling (SIR) particle filter is also introduced as the improved particle filter. From the simulation results using Matlab.

Researcher regularly generate statistical model based on past data in conventional way to predict flood water level so the particle filter use for estimation and tracking of nonlinear and dynamic systems is presented

SIS Particle Filter Algorithm

Step 1: Particle Generation

For time steps n=0,1,2,...

Initialization : for i=1, ..., N_P , draw the samples

$$x_n^{(i)} \approx q(x_n / x_{0:n\text{-}1}^{(i)}, y_{0:n}) \tag{2.1}$$

$$x_{0:n}^{(i)} = \left\{ x_{0:n\text{-}1}^{(i)}, x_n^{(i)} \right\} \tag{2.2}$$

Step 2: Weight computation

For i=1, ..., N_P , calculate the importance weights $W_n^{(i)}$

Step 3: Weight normalization

For i=1, ..., N_P, normalize the importance weights $\widetilde{W}_n^{(i)}$

Step 4: State estimation

For i=1, ..., N_P, calculate the state estimation.

SIR Particle Filter Algorithm

Step 1: Particle Generation

For time steps n=0,1,2,...

Initialization: for i=1,..., N_P (number of particles), draw the sample;

$$x_0^{(i)} \approx p(x_0), W_0^{(i)} = \frac{1}{N_p} \tag{2.3}$$

Step 2: Importance Sampling

For i=1,..., N_P, draw samples

$$\begin{aligned} \hat{x}_n^{(i)} &\approx p(x_n / x_{n\text{-}1}^{(i)}) \\ \hat{x}_{0:n}^{(i)} &= \left\{ x_{0:n\text{-}1}^{(i)}, \hat{x}_n^{(i)} \right\} \end{aligned} \tag{2.4}$$

$x_n^{(i)}$ the i-th simulated sample(particle)

Step 3: Weight Computation

Weight update: Calculate the importance weights;

$$W_n^{(i)} = p[y_n / \hat{x}_n^{(i)}] \tag{2.5}$$

Step 4: Weight normalization

Normalize the importance weights;

$$\widetilde{W}_n^{(i)} = \frac{w_n^{(i)}}{\sum_{j=1}^{N_p} W_n^{(j)}} \tag{2.6}$$

Step 5: Resampling algorithm

Resampling; Generate N_P newer particles $x_n^{(i)}$ from the set $\hat{x}_n^{(i)}$ according to the importance weights $\widetilde{W}_n^{(i)}$

Step 6: Repeat steps 2 to 5

The water level data is measured using Supervisory Control and Data Acquisition System (SCADA)

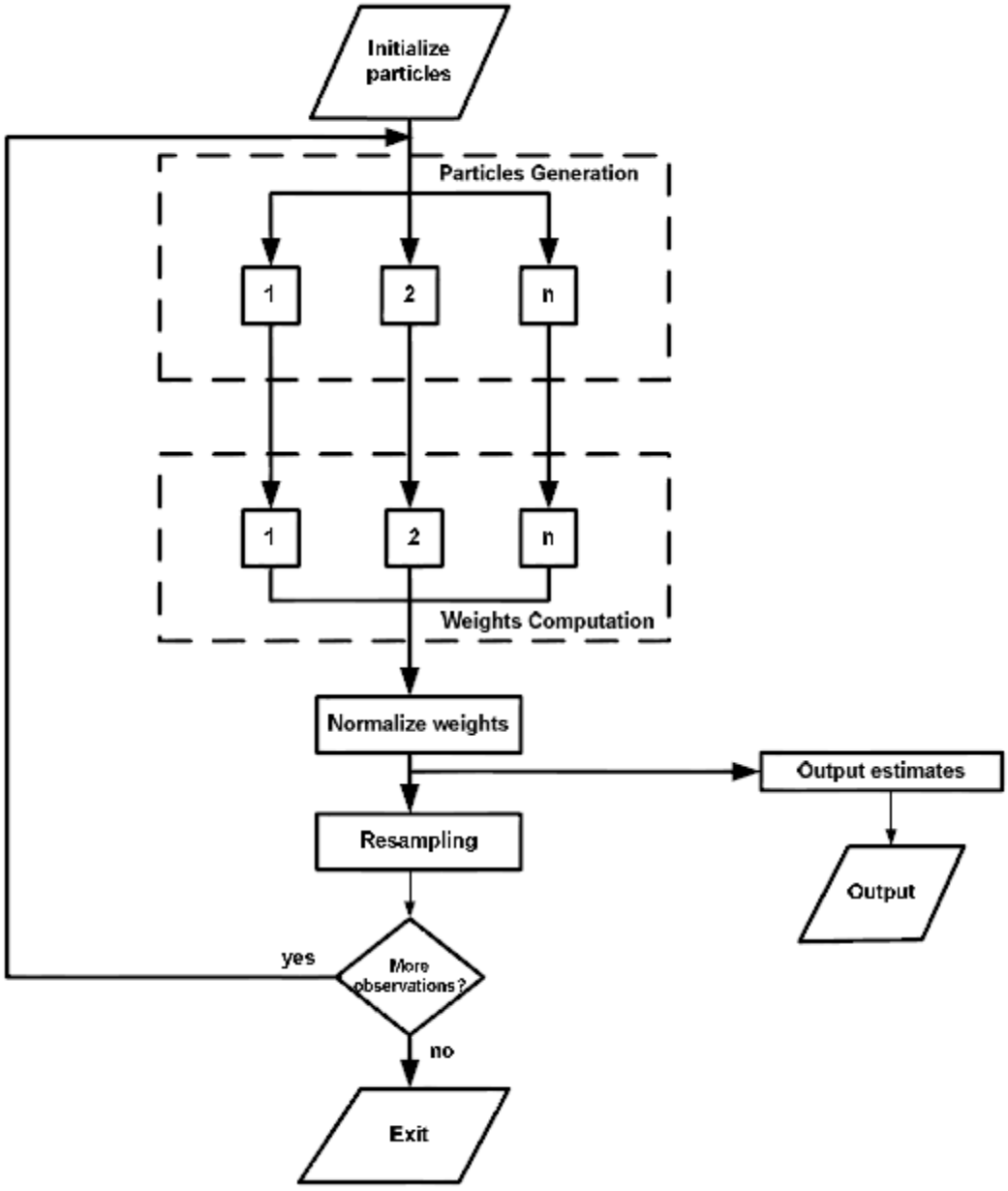


Figure2.1 Flow chart of particle filter [1]

The SIS and SIR particle filter can estimate the states well with larger number of particles but the simulation is only repeated for 150 times since longer runs produce much higher RMSE. The RMSE of estimation is applied to show the accuracy of filtering

$$RMSE(\hat{x}_n) = [\frac{1}{N_p} \sum_{i=1}^{N_p} (\hat{x}_n^{(i)} - x_n^{(i)})]^{1/2} \quad (2.6)$$

To verify the effectiveness of the particle filter algorithm, SIS particle filter and SIR particle filter algorithm are compared and analyzed.

The result of water level prediction using SIS particle filter algorithm is producing RMSE of 3.6994.

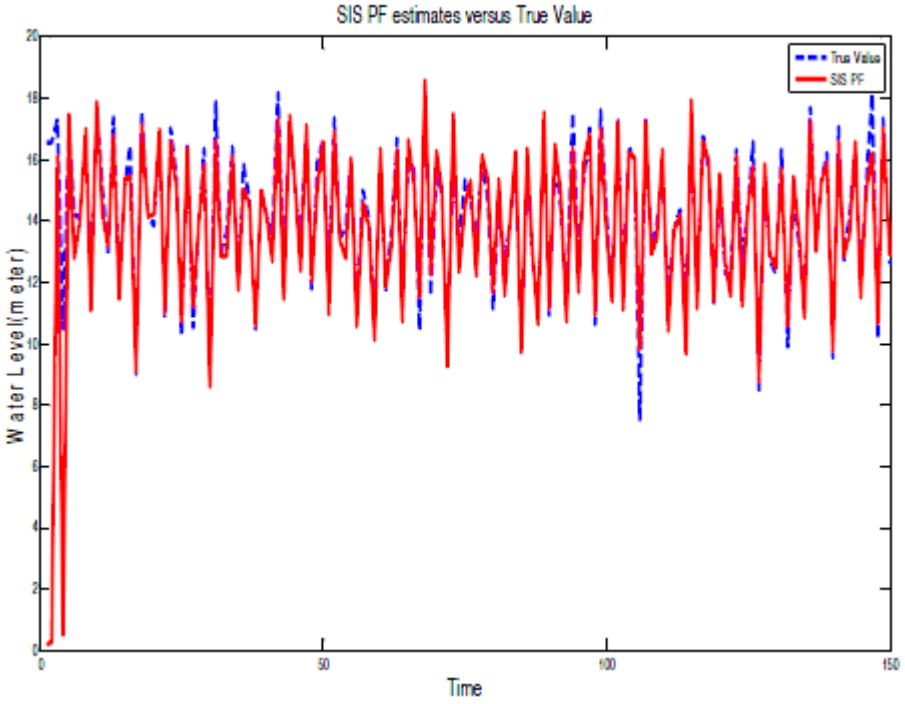


Figure 2.2 Water level prediction using SIS particle filter Algorithm [1]

The result of water level prediction using SIR particle filter algorithm is producing RMSE of 2.2202.

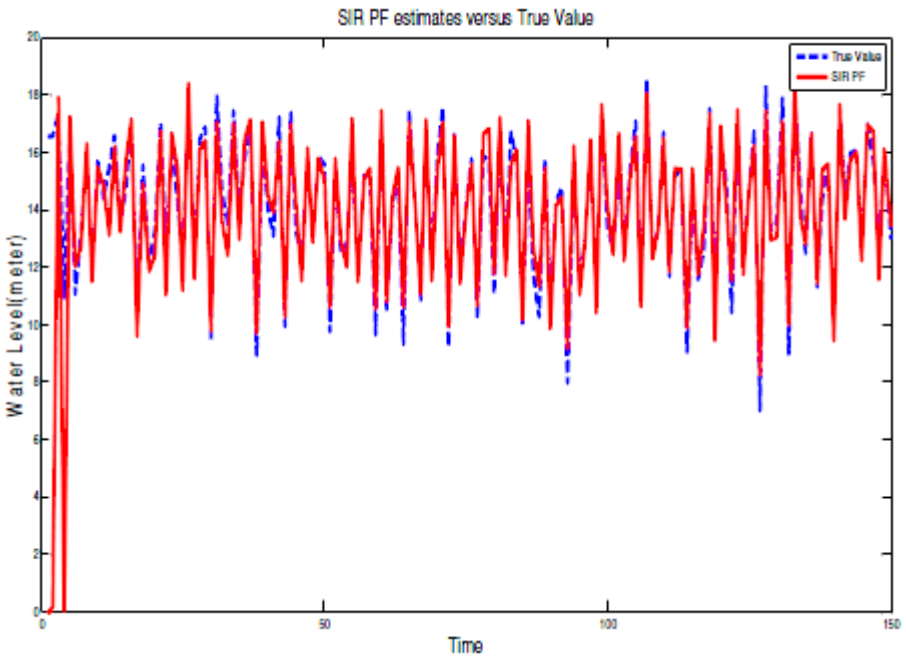


Figure 2.3 State estimation of SIR particle filter algorithm [1]

This study has demonstrated that SIR particle filter can be employed successfully to flood water level prediction and tracking problem. The SIR particle filter has an advantage of resampling method which will reduce the effect of particle degeneration problem that occurs in SIS particle filter.

2.2 Particle Filters for Positioning, Navigation and Tracking

"Particle Filters for Positioning, Navigation and Tracking" [2] is presented by Fredrik Gustafsson, Fredrik Gunnarsson, Niclas Bergman, Urban Forssell, Jonas Jansson, Rickard Karlsson, Per-Johan Nordlund this article use the particle filter based sequential Monte Carlo methods to develop tracking object application compare position with GPS. It consists of a class of motion models and a general non-linear measurement equation in position. For the improvement in accuracy, application apply Kalman filter use of non-linear models and non-Gaussian noise is the main explanation.

Motion models are related to position, velocity and acceleration which are linear in the state dynamics and non-linear in the measurements are considered:

Motion model : $$x_{t+1} = Ax_t + B_u u_t + B_f f_t \quad (2.7)$$

Measurement equations : $$y_t = hx_t + e_t \quad (2.8)$$

Particle Filter Implementation

- Generate sample of the state vector is referred to as a particle
- Update the weights by the likelihood
- Re-sampling: Bayesian bootstrap or Importance sampling base on the effective number of samples.
- Predict
- Iterate update the weights by the likelihood.

Application can track the object position and predict future positions. The simulation environments show a clear improvement in performance compared to existing Kalman filter based solutions. A choice of state coordinates making the state equation linear is beneficial for computation time

2.3 Maximum Likelihood Parameter Estimation in General State-Space Models using Particle Methods

"Maximum Likelihood Parameter Estimation in General State-Space Models using Particle Methods" [3] is presented by George Poyiadjis, Arnaud Doucet and Sumeetpal S. Singh. The article present new numerical methods to approximate the derivative of the optimal filter to perform batch and recursive maximum likelihood and tracking by maximizing the likelihood

The majority of the proposed particle filter SMC-based parameter estimation methods rely on augmenting the hidden state to include the unknown parameter and casting the problem they cannot be properly approximated using SMC methods, for a fixed number of particles, and the sufficient statistics degrade over time due to error accumulation

They present an original maximum likelihood method that is based on a direct particle approximation of the derivative of the optimal filter. The methods are based on the sequence of marginal distributions. They use the filter derivative approximation to compute the log-likelihood gradient and we combine it with a gradient ascent algorithm to generate maximum likelihood estimates of the model parameters.

Optimal Filter and its Derivatives

- State-Space Models

  This class of models includes many nonlinear and non-Gaussian time series models such as

$$x_{n+1} = \phi_\theta(X_n, +V_{n+1}), Y_n = \psi_\theta(X_n W_n) \quad (2.9)$$

- Optimal Filter Derivatives

The hidden process X_n is typically based on the sequence of joint posterior distributions $p_\theta(X_{0:n} \mid Y_{0:n})$ This summarizes all the relevant information available about $X_{0:n}$, up to time n. Using an importance sampling approach with an arbitrary important density $q_\theta(x_n \mid Y_n, x_{n-1})$; whose support includes the support $g_\theta(Y_n \mid x_n,) f_\theta(x_n \mid x_{n-1})$ it can be easily shown that the joint posterior density satisfies the recursion

$$p_\theta(X_{0:n} \mid Y_{0:n}) = \frac{\alpha_\theta(x_{n\text{-}1:n}, Y_n)}{p_\theta(Y_n \mid Y_{0:n\text{:-}1})} q_\theta(\mathrm{x_n} \mid Y_n, x_{n\ 1}) p_\theta(x_{0:n\text{:-}1} \mid Y_{0:n\text{:-}1}) \quad (2.10)$$

They are interested in the marginal $p_\theta(X_0 Y_{0:n})$, which is known as the filtering density. They will consider the first two derivatives of the optimal filter with respect to θ , namely $\nabla p_\theta(X_0 \mid Y_{0:n})$ and $\nabla^2 p_\theta(X_0 \mid Y_{0:n})$ .To simplify the notation, let main objective in this paper is to derive particle methods to approximate $\nabla p_\theta(X_0 \mid Y_{0:n})$ and $\nabla^2 p_\theta(X_0 \mid Y_{0:n})$

$$\nabla p_\theta(x_n \mid Y_{0:n}) = \frac{\nabla\ \varepsilon_\theta(x_n \mid Y_{0:n})}{\int \varepsilon_\theta(x_n \mid Y_{0:n}) dx_n} - p_\theta(x_n \mid Y_{0:n}) \frac{\int \nabla\ \varepsilon_\theta(x_n \mid Y_{0:n}) dx_n}{\int \varepsilon_\theta(x_n \mid Y_{0:n}) dx_n} \quad (2.11a)$$

$$\nabla^2 p_\theta(x_n \mid Y_{0:n}) = \frac{\nabla^2 \varepsilon_\theta(x_n \mid Y_{0:n})}{\int \varepsilon_\theta(x_n \mid Y_{0:n}) dx_n} - 2\nabla p_\theta(x_n \mid Y_{0:n}) x \frac{\int \nabla\ \varepsilon_\theta(x_n \mid Y_{0:n}) dx_n}{\int \varepsilon_\theta(x_n \mid Y_{0:n}) dx_n} - p_\theta(x_n \mid Y_{0:n}) \frac{\int \nabla^2\ \varepsilon_\theta(x_n \mid Y_{0:n}) dx_n}{\int \varepsilon_\theta(x_n \mid Y_{0:n}) dx_n} \quad (2.11b)$$

Particle Methods for the Filter Derivatives

- Particle Filters

Particle methods are used to numerically approximate the filtering recursion by means of a weighted empirical distribution of a set of N. They further assume that this weighted particle set is distributed approximately according to the joint density $p_\theta(x_{0:n-1} \mid Y_{0:n-1}$ A standard way to approximate the joint density at the next time step is to extend the path Sampling is achieved by first sampling the discrete index i using a standard resampling algorithm

Updated empirical distribution:

$$\hat{p}_\theta(x_{0:n} \mid Y_{0:n}) = \sum_{i=1}^{N} \tilde{a}_n^{(i)} \delta(x_{0:n} \mid X_{0:n}^{(i)})$$

$$\tilde{a}_n^{(i)} = \alpha_\theta(X_{n-1}^{\psi(i)}, X_n^{(i)}, Y_n) and \quad \tilde{a}_n^{(i)} = \frac{a_n^{(i)}}{\sum_{j=1}^{N} a_n^{(i)}} \tag{2.12}$$

Filter derivative approximations

The filter derivatives can take positive and negative values and integrate to zero. Empirical approximations of these measures using particle methods are still possible but with different weights. Computes the particle approximations approximate $\nabla p_\theta(X_0 \mid Y_{0:n})$ and $\nabla^2 p_\theta(X_0 \mid Y_{0:n})$ by propagating the weighted particles on the path space and marginalizing the expressions

- Particle algorithm

They describe the proposed sequential method to approximate the first two derivatives of the optimal filter

As in the standard particle filter, we generate a set of particles $x_n^{(i)}$ for i = 1, ..., N, Evaluating the pointwise approximations at points $x_n^{(i)}$ yields the following particle approximations

ML Parameter Estimation

Now consider the general state space model assume that the model that generates the observation sequence Y_n ; n>=0 they propose here two gradient algorithms to perform maximum likelihood estimation. These are based on a gradient ascent method that utilizes the estimates of the derivatives of the filter that were presented in the previous section.

- Recursive ML

A standard approach to Recursive ML (RML) estimation considers a series of log-likelihood functions

$$\log p_\theta(Y_{0:k}) = \sum_{n=0}^{k} \log p_\theta(Y_n \mid Y_{0:n-1}) \tag{2.13}$$

The expression $p_\theta(Y_n \mid Y_{0:n-1})$ is known as the predictive likelihood and can be written as

$$p_\theta(Y_n \mid Y_{0:n-1}) = \iint g_\theta(Y_n \mid x_n) f_\theta(x_n \mid x_{n-1}) p_\theta(x_{n-1} \mid Y_{0:n-1}) dx_{n-1:n}$$

$$\nabla \log \hat{p}_\theta(Y_n \mid Y_{0:n-1}) = \frac{\sum_{j=1}^{N} \rho_n^{(j)}}{\sum_{j=1}^{N} a_n^{(j)}} \tag{2.14}$$

- Adaptive SA

The Hessian of the log-likelihood can be straightforwardly estimated using the particle approximations of the optimal filter and its first and second derivatives

$$\nabla^2 \log \hat{p}_\theta(Y_n \mid Y_{0:n-1}) = \frac{\sum_{j=1}^{N} \pi_n^{(j)}}{\sum_{j=1}^{N} a_n^{(j)}} - \left(\frac{\sum_{j=1}^{N} \rho_n^{(j)}}{\sum_{j=1}^{N} a_n^{(j)}}\right)^2 \tag{2.15}$$

This adaptive SA is particularly attractive, in terms of convergence acceleration

- RML Parameter Estimation Algorithm

  The Recursive ML estimation is summarized as follows:

  1. Sampling Step
  2. Weight Calculation
  3. Parameter Update Step

- Batch ML

In cases where a set of observations $Y_{0:n}$ is available, we describe here a batch version (BML) of the previous algorithm. This algorithm maximizes the log-likelihood $\log p_\theta(Y_{0:n})$

$$\nabla \log \hat{p}_{\theta_{m-1}}(Y_{0:n}) = \sum_{k=0}^{n} \frac{\sum_{j=1}^{N} \rho_k^{(j)}}{\sum_{j=1}^{N} a_k^{(j)}} \tag{2.16}$$

Numerical study

The RML and BML algorithms were tested, based on artificial and real observations.

- Linear Gaussian State Space Model

$$\begin{aligned} x_{n+1} &= \delta X_n + \sigma_v V_{n+1}, X_0 \approx \quad N(0, \frac{\sigma_V^2}{1-\delta^2}) \\ Y_n &= X_n + \sigma_w W_n \end{aligned} \tag{2.17}$$

In such a model, the optimal filter is given by the Kalman filter and exact expressions for the first and second derivative of the filter can be obtained.so they compare their numerical methods with basic methods

The comparison were almost indistinguishable

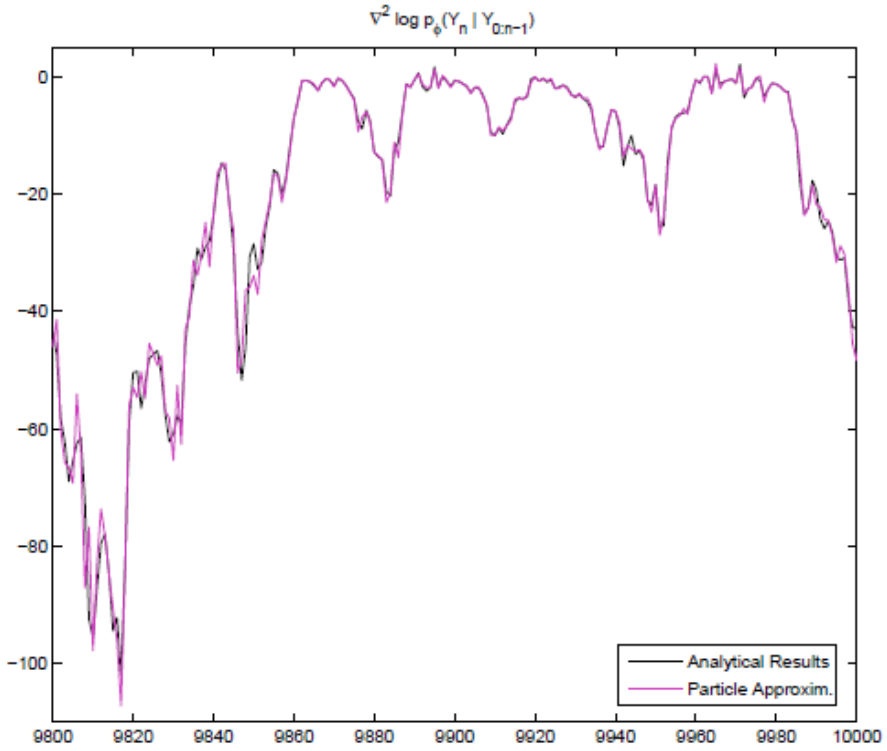


Figure 2.4 Analytical and numerical results for linear Gaussian state space model [3]

- Stochastic Volatility Model

The RML algorithm was implemented using the following Stochastic Volatility model

$$X_{n+1} = \delta X_n + \sigma_v V_{n+1}, X_0 \approx \quad N(0, \frac{\sigma_V^2}{1-\delta^2}) \qquad (2.18)$$
$$Y_n = \beta \exp(X_n/2) W_n$$

- Parameter tracking

A standard approach to track a time-varying parameter is to set the step-size to a small positive number γ An example of the tracking performance of the RML algorithm based on the linear Gaussian state space model, having a time-varying drift parameter δ

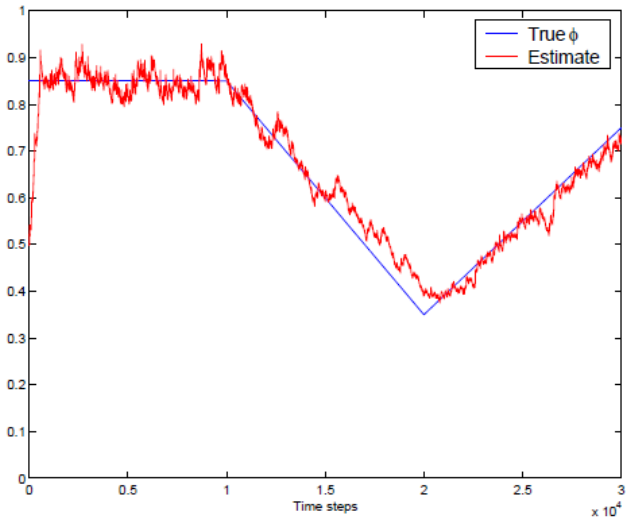


Figure 2.5 RML algorithm tracking performance for time varying δ [3]

This paper has presented original particle methods to estimate the first and second derivative of the optimal filter in general state-space models. The methods use non-standard particle methods to approximate the Hahn-Jordan decomposition of the resultant signed measures.

Based on this, they propose a recursive and a batch algorithm to perform ML parameter estimation using a gradient ascent method. The Hessian estimate can be used as an adaptive step-size in the gradient ascent recursion to provide faster convergence of the algorithm.

The proposed methods can also be extended to the case where it is possible to integrate analytically a subset of the state variables.

2.4 Particle Filter Positioning and Tracking Algorithm based on Dynamic Model

"Particle filter positioning and tracking algorithm based on dynamic model" [4] is presented by TIAN-Zengshan, LUO Lei This method can apply to any state-space model which is nonlinear system, and the accuracy can approach to best of all. This particle filter method new Monte Carlo particle filter method can be used effectively to inhibit Non-line-of-sight (NLOS) errors and can be combined with positioning and tracking model to get higher precision.

Location estimation of mobile target is singer model. It use to that acceleration a(t) is regarded as zero mean random process which has index autocorrelation, the process can be modeled as follows:

$$\mathrm{R}(\tau(= \mathrm{E}[\mathrm{a}(\mathrm{t})\mathrm{a}(\mathrm{t}+\tau)] = \sigma_m^2 e^{-\alpha|\tau|} \tag{2.19}$$

So the state function of the continuous target motion can be obtained as follows

$$\dot{\mathrm{X}}(\mathrm{t}) = \mathrm{AX}(\mathrm{t}) + \widetilde{\mathrm{W}}(\mathrm{t}) \tag{2.20}$$

$$\mathrm{X}(\mathrm{t}) = \left[x(\mathrm{t}), \dot{x}(\mathrm{t}), y(\mathrm{t}), \dot{y}(\mathrm{t}), \ddot{y}(\mathrm{t})\right]^T \tag{2.21}$$

$x(\mathrm{t}), \dot{x}(\mathrm{t})$ denotes the position and the speed of the goal in X direction separately $y(\mathrm{t}), \dot{y}(\mathrm{t}), \ddot{y}(\mathrm{t})$ denotes the position, the speed and the acceleration of the goal in Y direction separately; $\widetilde{\mathrm{W}}(\mathrm{t})$ is Gaussian white noise which mean is 0, Variance is $2\alpha\sigma_m^2$

Discretize the function with sampling gap T:

The transition matrix is:

$$F = \begin{bmatrix} \begin{bmatrix} 1 & T & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & T & (\alpha\alpha - 1 + \mathrm{e}^{-\alpha\mathrm{T}})/\alpha^2 \\ 0 & 0 & 0 & 1 & (1 - \mathrm{e}^{-\alpha\mathrm{T}})/\alpha \\ 0 & 0 & 0 & 0 & \mathrm{e}^{-\alpha\mathrm{T}} \end{bmatrix} \end{bmatrix} \tag{2.22}$$

The covariance matrix of the state noise is:

$$Q(k) = \begin{bmatrix} 2\alpha\sigma_m^2 \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & q_{11} & q_{12} & q_{13} \\ 0 & 0 & q_{21} & q_{22} & q_{23} \\ 0 & 0 & q_{31} & q_{32} & q_{33} \end{bmatrix} \end{bmatrix} \tag{2.23}$$

$$q_{11} = \frac{1}{2\alpha^5}(1 - e^{-2\alpha T} + 2\alpha T + \frac{2\alpha^3 T^3}{3} - 2\alpha^2 T^2 - 4\alpha T e^{-\alpha T})$$

$$q_{12} = q_{21} = \frac{1}{2\alpha^4}(e^{-2\alpha T} + 1 - 2e^{-\alpha T} + 2\alpha T e^{-\alpha T} - 2\alpha T - \alpha^2 T^2)$$

$$q_{13} = q_{31} = \frac{1}{2\alpha^3}(1 - e^{-2\alpha T} - 2\alpha T e^{-\alpha T})$$

$$q_{22} = \frac{1}{2\alpha^3}(4e^{-\alpha T} - 3 - e^{-2\alpha T} + 2\alpha T)$$

$$q_{23} = q_{32} = \frac{1}{2\alpha^2}(e^{-2\alpha T} + 1 - 2e^{-\alpha T})$$

$$q_{33} = \frac{1}{2\alpha}(1 - e^{-2\alpha T}) \quad (2.24)$$

TDOA OBSERVATION EQUATION

$$h(X(k)) = \begin{bmatrix} \sqrt{(x(k)-x_2)^2 + y(k)-y_2)^2} \\ \sqrt{(x(k)-x_3)^2 + y(k)-y_3)^2} \\ \cdots \\ \sqrt{(x(k)-x_n)^2 + y(k)-y_n)^2} \end{bmatrix} \quad (2.25)$$

There are mainly two measurement noise errors

- Measurement error

Measurement error is Gaussian random variables that the mean is 0 m and the standard deviation is between 20 to 60 m

- NLOS error

NLOS error obey the index distribution model that the mean is 650 m and the standard deviation is 0-1200 m. NLOS error can be generated by the distribution of power delay, and the Probability Density Function caused by the excessive delay of NLOS

$$D(\iota) = \begin{cases} \frac{1}{C\tau_{\text{rms}}} \exp(-\frac{1}{C\tau_{\text{rms}}}) & d > 0 \\ 0 & d \le 0 \end{cases} \quad (2.26)$$

THE BASIC PRINCIPLE OF PARTICLE FILTER

Suppose:

$X_{0:k} = X(0), X(1), \ldots, X(k), Y_{1:k} = Y(1), Y(2), \ldots, Y(k)$ the probability distribution of a certain time in 0 to k, $p(X_{0:k})$

$$p(X_{0:k}) = \left\{ X_{0:k}^{(m)}, \omega_k^m \right\}_{m-1}^{M} \tag{2.27}$$

So the posterior probability density in t moment can be approximately denoted by discrete weights:

$$p(X_{0:k} | Y_{1:k}) \approx \sum_{m-1}^{M} \omega_k^{(m)} \, \delta\left\{ X_{0:k}, X_{0:k}^{(m)} \right\} \tag{2.28}$$

In order to test the performance of algorithm, three base stations which coverage radius is five kilometers are used for TDOA location simulation the initial position is (2000 m, 1000 m) and the velocity of horizontal direction is 10 m/s. The initial velocity is 5 m/s and the initial acceleration is 0 in vertical direction.

We can see that particle filter has good performance on NLOS error both 100 and 20 groups and make the good match with the real track.

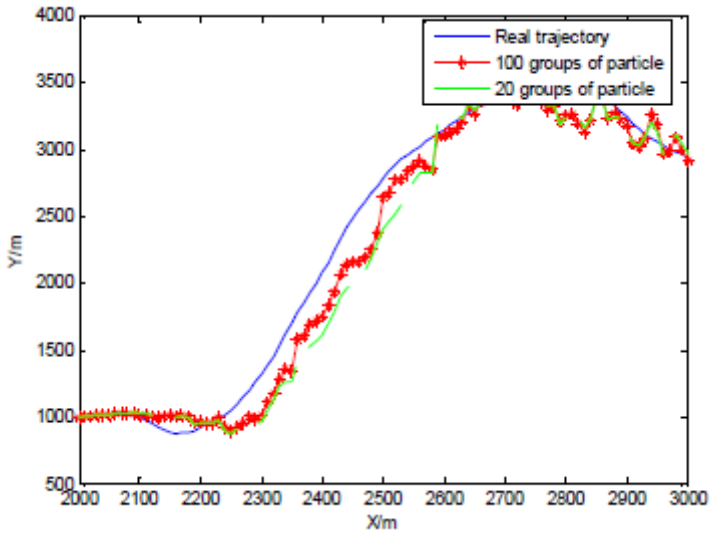


Figure 2.6 Comparison of tracking results for different particles [4]

2.5 Moving Target Tracking via article Filter Based on Color and Contour Features

In [5] “Moving Target Tracking via article Filter Based on Color and Contour Features” is presented by Jiayan Li, Xiaofeng Lu, Lianyi Ding and Hengli Lu

This research found that only color base cannot detect object in accuracy when background is complex thus, they apply contour information of a moving target from its motion segmentation in the application.

Contour is another important feature, it could reflect the basic shape of a target. Therefore the calculation is complex and the processing speed is slow. However, motion segmentation could also help to extract contours. The extracted contour through motion segmentation is “motion contour” it is suitable to represent a moving target and also can eliminate influences of the background to some extent.

Particle Filter Algorithm

$$x_t = x_{t\text{-}1} + (x_{t\text{-}1} x_{t\text{-}2}) + n_t \tag{2.29}$$

Donates n_t : Gaussian noise

Color Model and Measurement

Color histogram is used to describe the color feature.

$$p_u(y) = C \sum_{x_i \in R} k_c \left(\frac{\|x_i - y\|}{d} \right) \delta[b(x_i) - u] \tag{2.30}$$

A popular similarity measurement between two color distributions is the Bhattacharyya Coefficient

$$\rho[p,q] = \sum_{u=1}^{m} \sqrt{p_u(y_0) q_u(y_t)} \tag{2.31}$$

The larger ρ is, the more similar the two color histogram. The associated Bhattacharyya distance is

$$d_{color} = \sqrt{1 - \rho[p,q]} \tag{2.32}$$

The particle's weight updated by d_{color}

$$w_{color}^{i} = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{d_{color}^{2}}{2\sigma^{2}}) \quad (2.33)$$

Contour Model

The aim of contour extraction is to obtain the shape characteristics of a target and contours of other objects in the background do not contribute to the tracking algorithm, frame difference method is adopted here. Frame difference takes a subtraction between the adjacent frames. The appearances of a moving target are always accompanied by motion, the regions could reflect the shapes of the target but that not accurate enough so they adopt Canny edge detector and by a logic AND operation of the subtracted image with the Canny edge image, the motion edge image is obtained. The motion edge is not the motion contour, in order to acquire the continued motion contours, connected domain detector should be applied to the motion edge image. Choose a set of feature points to represent the contours. The feature points lie in the lines of contours, but they also lie in the lines of edges.

Harris corner detector is used to abstract feature points. The gained points are well-proportioned rotation or zooming. Harris operator is realized by calculating each pixel's gradient then judge the pixel as a corner.

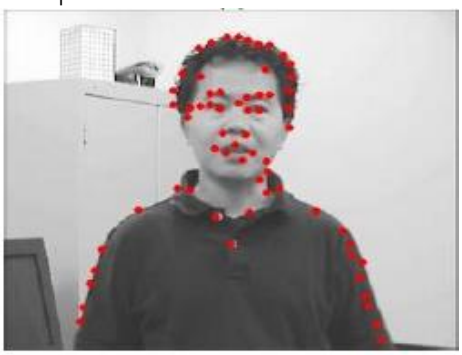


Figure 2.7 Result of Harris corner detection [5]

In tracking approach, each particle represents a candidate object. The measurement is to compare the degree of similarity between the candidate's contour points and the target's contour points. The comparative result is the basis to judge the weight value of a particle.

Use Hausdorff distance to compare between 2 set of point, the result call “partial Hausdorff distance” $d_{contour}$ The smaller $d_{contour}$ is, the more similar the candidate’s contour is to the target’s contour

$$w_{contour}^{i} = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{d_{contour}^{2}}{2\sigma^{2}}) \quad (2.34)$$

Fusion algorithm flow

- State predicting

  Predict the new state of particles according to a predefined dynamic model

- Weight updating

  Extract the color feature by establishing the color histogram. Measure the Bhattacharyya distance with the target’s color histogram. Update the weight of color

  Extract the contour feature by establishing the contour point set. Measure the Hausdorff distance with the target’s contour. Update the weight of contour.

- Feature fusing

  The fusion scheme is to integrate the weight of color and the weight of contour into a particle’s weight.

$$w_{t}^{i} = (1-\alpha) w_{t(color)}^{i} + \alpha w_{t(contour)}^{i} \quad (2.35)$$

- Estimated state output

  Calculate the estimated state and output the result.

- Resampling

  Propagate samples from the particles with bigger weights and eliminate the particles with smaller weights.

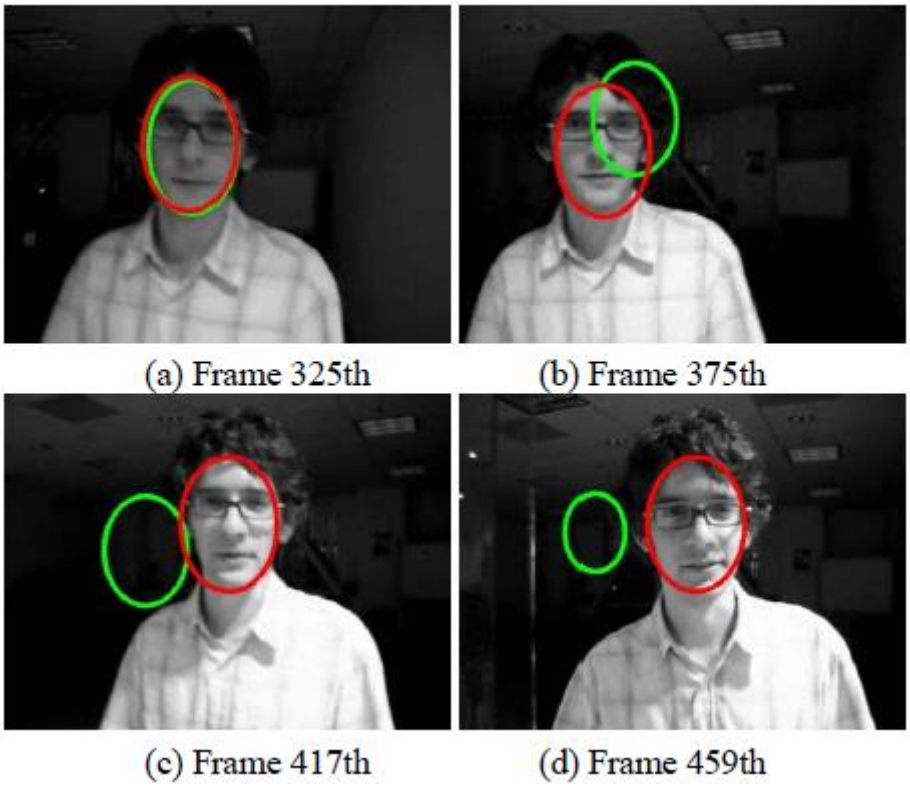


Figure 2.8 Tracking results under background of illumination changes [5]

The algorithm can track the target robustly and accurately in the condition of illumination changes and similar color clutters.

2.6 Particle Filter Tracking Based on Color and SIFT Features

In [6] "Particle Filter Tracking Based on Color and SIFT Features" is presented by Peiliang Wu, Lingfu Kong, Fengda Zhao, Xianshan Li. This paper, particle filter algorithm is proposed in which the weight of particle is determined by both the color cue and local scale invariant feature transform (SIFT) features.

The particle weight is calculated firstly by color similarity measurement and then updated according to the distribution of SIFT matches.

The particle filter tracking based on color is the same as [5] and this paper apply SIFT in the application

Scale invariant feature transform

SIFT algorithm selects the key points at maxima and minima of a difference of Gaussian function applied in scale space to achieve rotation invariance

The proposed method can effectively improve the tracking precision especially when the object is scale changing or in the clutter background of similar colors. The orientation is determined by the peak in a histogram of local gradient orientations. The SIFT feature descriptor is represented as f ={p, s, o, hist} where p is the 2-D position of the feature in terms of the image coordinate, s is the feature scale, o is the feature vector direction, and hist is the gradient orientation distribution quantized into 128 bins.

SIFT-based object recognition is performed by matching each key point extracted from the current image independently to SIFT model built offline. The best candidate match for each key point is found by identifying its first nearest neighbor

Hybrid PF based on color and SIFT

- Outliers discarding

  So we firstly discard the outliers of SIFT matches using the RANSAC algorithm

- Particle weight allocation

  They combine the local SIFT feature distributions as a distance kernel function to update the weights of particles and pull them back around the object center.

  The weight of each particle is allocated as follows:

  Step1. Approximate the object center O(x, y) by

$$x = \sum_{i=1}^{M} x_i / M, y = \sum_{i=1}^{M} y_i / M \tag{2.36}$$

  Step2. For n=1,...,N

  (a) Count the Euclidean distance d_n from $p_n^{(t)}$ to O.

  (b) Obtain the factors using exponential kernel function

$$k_s(d_n) = \begin{cases} 1 & d_n \le T_d \\ \exp\left[(t_d - d_n) / \max\{d_i - T_d\}_{i-1}^{N}\right] & d_n > T_d \end{cases} \tag{2.37}$$

  (c) Update the weight $\omega_n^{(t)}$ using

$$\omega_n^{(t)} = k(d_n) \cdot \omega_n^{(t)} \tag{2.38}$$

Step3. Uniform the weights

$$\varpi_n^{(t)} = \omega_n^{(t)} \Big/ \sum_{i=1}^{N} \omega_n^{(t)}, n = 1,....., N \qquad (2.39)$$

- Tracking window modification

Tracking window is described as an ellipse region with long and short axis to cover the tracked object. To keep precise tracking, the size of tracking window should be modified automatically. Considering the size of color region and SIFT matches distribution region in current frame.

- Total flow of processing

The flow of the processing at time t is as follows.

Step1. Select and propagate samples at time t from the samples at time t-1 according to their weights.

Step2. Predict the state of the samples at time t according to the predefined dynamic model.

Step3. Calculate the weights of the samples according to the color Bhattacharyya distance.

Step4. Find matches between SIFT model and tracking window ht. If matches quantity is larger than threshold, update weights of particles.

Step5. Calculate the mean state.

Step6. Modify size of tracking window.

Figure 2.9 Ink box tracking use color-based particle filter (left) and hybrid Color-SIFT particle filter (right). [6]

These processes are iterated for every image frame, by which the object is tracked. In this paper they extract and match SIFT key points in the whole image area, while the

extracting and matching could also be executed only in the regions constraint by color cues to improve the tracking speed.

2.7 Transductive Inference for Color-Based Particle Filter Tracking

In [7] Jiang LI and Chin-Seng CHUA present “Transductive Inference for Color-Based Particle Filter Tracking” this paper present an adaptive color-based particle filter tracking algorithm by transductive inference. The particles are given different weights as weighted unlabeled data by a weak classifier. Combining confidently labeled data and weighted unlabeled data

They fit the transductive learning into the framework of Monte Carlo-based particle filter and relax the Gaussian and linear assumptions. Combining both labeled data and weighted particles, the proposed transductive particle filter algorithm can adapt the object color model effectively.

Non-linear non-Gaussian Bayesian Tracking

The posterior density $p(X_k|Y_k)$ and the observation density $p(X_k|Y_k)$ are often non-Gaussian and multi-modal. The key idea of particle filtering is to approximate the probability distribution by a weighted sample set

The target region is represented by a bounding box and the sample set is propagated through the application of a dynamic model $X_k=Ax_{k-1}+U_{k-1}$

They use color histograms to represent the color distributions of target objects. The color histograms are calculated in HS color space using 8 x 8 bins.

The pixels in the outlying areas of the target have a higher probability to be considered as clutter than the inner part.

The weighting function is defined $k(r)=1r^2$;r<1 and =0; r>=1 where T is the distance from the region centroid. Thus, we increase the target model's robustness against outlier clutter.

Z_i be the pixel locations of the target candidate, centered at y at the current image. The color distribution:

$$p(y)^{(u)}= \oint \sum_{i=1}^{n_h} k\left(\frac{\|y- z_i\|}{a} \right)\delta[h(z_i)-u] \qquad (2.40)$$

Likelihood modeling is difficult mainly because of the complexity of the relationship between the tracker's state and the image. They use color histogram intersection to model the state likelihood $p(y_t \mid x_t)$

$\rho[p,q] = \frac{\sum \min(p,q)}{\sum q}$ is weighing the larger ρ is the more similar between two color histograms. A distance between two color distributions can be defined the same as $d = \sqrt{1-\rho[p,q]}$ The likelihood probability of each sample can be counted as $W^n = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{d^2}{2\sigma^2}}$ A Transductive Inference. They divide confident label in two parts : labeled samples $L = \{(s_j, \iota_j), j = 1,\ldots., N_L\}$ and unlabeled samples $U = \{s_j, j = 1,\ldots., N_U\}$

Where N_L and N_U are the size of the labeled samples and unlabeled samples respectively

The transductive classification can be written as:

$$\iota_i = \arg \max_{j=1,\ldots.,C} p(\iota_i \mid s_j, L, U : \forall s_j \in U) \quad (2.41)$$

The classification confidence can be calculated based on the likelihood function assigned by this weak classifier

$$\omega_q(j) = \omega_{k-1}^j p(y_k \mid \chi_k^j) \quad (2.42)$$

The color model adaptation will be performed on the new weighted data set

$$D' = L \cup \{s_j, \omega_q(j) : \forall s_j \in U\} \quad (2.43)$$

A Transduction Algorithm for Adaptive Color Tracking

The mean state histogram can be seen as the confidently labeled data. $\{p_s, j = 1,\ldots., N_s\}$ are the histograms of particles. $W_q(j)$ is used to weight unlabeled data.

$$q_k + 1 = (1 \quad \alpha \quad \beta) q_k + \alpha p_{E_{(S_k)}} + \beta \sum_{j=1,\ldots,N_s} p_{s_j} W_q(j) \quad (2.44)$$

Iteration Steps of the Transductive Color-Based Particle Filter Tracker

- Calculate the proposal probability
- Select N_s samples from the set x_{t-1}
- Calculate the likelihood function
    - Calculate the color distribution
    - Calculate the histogram interaction distance
    - likelihood function
- Assign each particle a weight
- Calculate total weights and get effective sample size N_{eff}
    - Normalize
    - calculate the effective sample size N_{eff}
- If N_{eff} is lower than a preset threshold, then resample.
- Estimate the mean state of the set X_K
- Color model adaptation by transductive learning
    - update object color model by transductive learning

The experiments show that the transductive particle filter algorithm can effectively update the object color model and maintain object tracking even under temporal occlusion and color clutter.

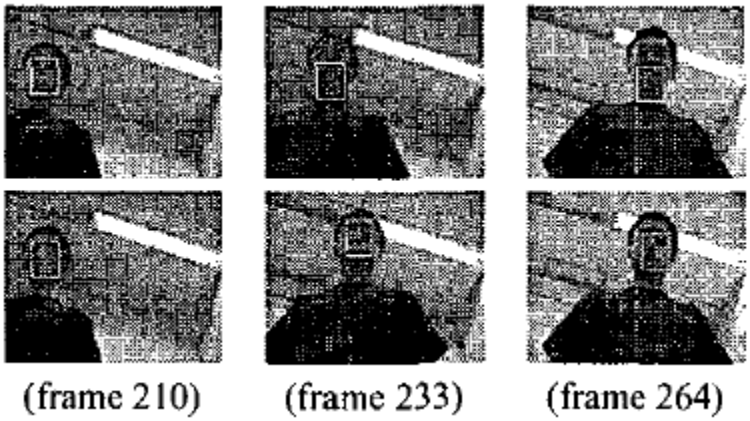


Figure 2.10 the upper row is the results of the normal particle filter algorithm and the bottom row is the results of the transductive particle filter algorithm [7]

CHAPTER 3
THEORY

3.1 Particle Filter

Particle filters or Sequential Monte Carlo (SMC) methods are a set of on-line posterior density estimation algorithms that estimate the posterior density of the state-space by directly implementing the Bayesian recursion equations. Particle filter methods use a grid-based approach, and use a set of particles to represent the posterior density. These filtering methods make no restrictive assumption about the dynamics of the state-space or the density function. Particle filter methods provide a well-established methodology for generating samples from the required distribution without requiring assumptions about the state-space model or the state distributions. The state-space model can be non-linear and the initial state and noise distributions can take any form required. However, these methods do not perform well when applied to high-dimensional systems. Particle filter methods implement the Bayesian recursion equations directly by using an ensemble based approach. The samples from the distribution are represented by a set of particles; each particle has a weight assigned to it that represents the probability of that particle being sampled from the probability density function.

Weight disparity leading to weight collapse is a common issue encountered in these filtering algorithms; however it can be mitigated by including a resampling step before the weights become too uneven. In the resampling step, the particles with negligible weights are replaced by new particles in the proximity of the particles with higher weights. The objective of a particle filter is to estimate the posterior density of the state variables given the observation variables. The particle filter is designed for a hidden Markov Model, where the system consists of hidden and observable variables. The observable variables (observation process) are related to the hidden variables (state-process) by some functional form that is known. Similarly the dynamical system describing the evolution of the state variables is also known probabilistically.

A generic particle filter estimates the posterior distribution of the hidden states using the observation measurement process

The objective of the particle filter is to estimate the values of the hidden states $\boldsymbol{Y}$, given the values of the observation process $\boldsymbol{Y}$.

The particle filter aims to estimate the sequence of hidden parameters, x_n for n = 0,1,2,3,..., based only on the observed data y_n for n = 1,2,3,.... All Bayesian estimates of x_n follow from the posterior distribution $p(x_n \mid y_1, y_2,...,y_n)$.

General state space model is

System model : $$x_n = f(x_{n-1}) + W_n) \tag{3.1}$$

Observation model : $$Y_n = G(Y_{n-1}) + V_n) \tag{3.2}$$

W_n denote the system noise ;

V_n denote the observation noise ;

Suppose $X_{0:n}$ {X = X(0),X(1),...,X(n)}

$Y_{1:n}$ {Y = Y(1),Y(2),...,Y(n)}, the probability distribution of a certain time in 0 to n, $P(X_{0:n})$ can be expressed as:

$$P(X_{0:n}) = \{X_{0:n}^{(i)}, w_n^{(i)}\}_{i-1}^{N} \tag{3.3}$$

$X_{0:n}^{(i)}$ denote the N particle of a certain time in 0 to n;

$w_n^{(i)}$ denote Normal weight of N particle;

$\sum_{-n=1}^{-N} w_n^{(i)}$ = 1 : N is the number of particles.

The posterior probability density in n moment can pre-approximately denoted by discrete weights:

$$P(X_{0:n} \mid Y_{1:n}) \approx \sum_{n=1}^{N} w_n^{(i)} \delta\{X_{0:n} - \mathrm{X}_{0:\mathrm{n}}^{(\mathrm{i})}\} \tag{3.4}$$

3.1.1 Density Estimation

Density estimation is the construction of an estimate, based on observed data, of an unobservable underlying probability density function. The unobservable density function is thought of as the density according to which a large population is distributed; the data are usually thought of as a random sample from that population.

A variety of approaches to density estimation are used, including Parzen windows and a range of data clustering techniques, including vector quantization. The most basic form of density estimation is a rescaled histogram.

3.1.2 Bayesian Recursive estimation

Bayesian filter, is a general probabilistic approach for estimating an unknown probability density function recursively over time using incoming measurements and a mathematical process model.

The true state X is assumed to be an unobserved Markov process, and the measurements Y are the observed states of a Hidden Markov Model (HMM). The following picture presents a Bayesian Network of a HMM.

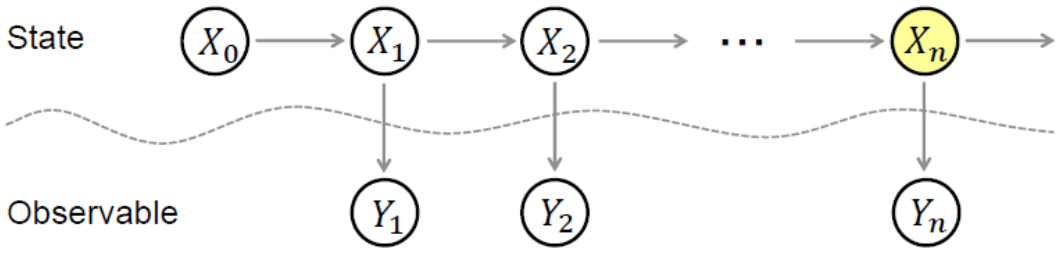


Figure 3.1 Bayesian Network of a HMM.

Because of the Markov assumption, the probability of the current true state given the immediately previous one is conditionally independent of the other earlier states. $P(X_n \mid X_{n\text{-}1})$

Similarly, the measurement at the n-th timestep is dependent only upon the current state, so is conditionally independent of all other states given the current state. $P(Y_n \mid X_n)$

However, when using the Kalman filter to estimate the state **X**, the probability distribution of interest is associated with the current states conditioned on the measurements up to the current timestep. This leads to predict and *update* steps of the Kalman filter written probabilistically. The probability distribution associated with the predicted state is the sum (integral) of the products of the probability distribution associated with the transition from the (*n* - 1)-th timestep to the *n*-th and the probability distribution associated with the previous state, over all possible $X_{n\text{-}1}$

$$P(X_n \mid Y_{1:n}) = \int \ldots \int P(X_{0:n} \mid Y_{1:n}) dX_{n-1} \ldots dX_0 \quad (3.5)$$

3.2 Likelihood

Likelihood is a function of how likely an event is, which is weaker than probability a likelihood function is a function of the parameters of a statistical model, defined as follows: the likelihood of a set of parameter values given some observed outcomes is equal to the probability of those observed outcomes given those parameter values. Likelihood functions play a key role in statistical inference, especially methods of estimating a parameter using statistics

For the normal distribution $N(u, \sigma^2)$ which has probability density function

$$f(x \mid u, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma}} \exp(-\frac{(x-u)^2}{2\sigma^2}) \quad (3.6)$$

The corresponding probability density function for a sample of independent identically distributed normal random variables (the likelihood) is

$$f(x_1, \ldots, x_n \mid u, \sigma^2) = (\frac{1}{2\pi\sigma^2})^{n/2} \exp(-\frac{\sum_{-i=1}^{-n} (x_i - u)^2}{2\sigma^2}) \quad (3.7)$$

Or more conveniently:

$$f(x_1, \ldots, x_n \mid u, \sigma^2) = (\frac{1}{2\pi\sigma^2})^{n/2} \exp(-\frac{\sum_{i=1}^{n} (x_i - \bar{x})^2 + n(\bar{x} - u)^2}{2\sigma^2}) \quad (3.8)$$

Where $\bar{x}$ is the sample mean.

This family of distributions has two parameters: $\theta = (u, \sigma)$, so we maximize the likelihood, $L(u, \sigma) = f(x_1, \ldots, x_n \mid u, \sigma)$ over the both parameters simultaneously, or if possible, individually.

Since the logarithm is a continuous strictly increasing function over the range of the likelihood, the values which maximize the likelihood will also maximize its logarithm. Since maximizing the logarithm often requires simpler algebra, it is the logarithm which is maximized below.

$$
\begin{aligned}
0 &= \frac{\partial}{\partial - u} \log\left(\frac{1}{2\pi\sigma^2}\right)^{n/2} \exp\left(-\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2 + n(\bar{x} - u)^2}{2\sigma^2}\right) \\
0 &= \frac{\partial}{\partial\ u} \log\left(\frac{1}{2\pi\sigma^2}\right)^{n/2} \left(-\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2 + n(\bar{x} - u)^2}{2\sigma^2}\right) \\
0 &= 0 - \frac{-2n(\bar{x} - u)}{2\sigma^2}
\end{aligned}
\tag{3.9}
$$

Which is solved by

$$\hat{u} = \bar{x} \sum_{n=1}^{n} (x_i / n) \tag{3.10}$$

This is indeed the maximum of the function since it is the only turning point in **u** and the second derivative is strictly less than zero. Its expectation value is equal to the parameter **u** of the given distribution, $E[\hat{u}] = u$, which means that the maximum-likelihood estimator $\hat{u}$ is unbiased.

Similarly we differentiate the log likelihood with respect to **σ** and equate to zero:

$$
\begin{aligned}
0 &= \frac{\partial}{\partial\sigma} \log\left(\frac{1}{2\pi\sigma^2}\right)^{n/2} \exp\left(-\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2 + n(\bar{x} - u)^2}{2\sigma^2}\right) \\
&= \frac{\partial}{\partial\sigma} \left(\frac{n}{2} \log\left(\frac{1}{2\pi\sigma^2}\right) - \frac{\sum_{i=1}^{n}(x_i - \bar{x})^2 + n(\bar{x} - u)^2}{2\sigma^2}\right) \\
&= -\frac{n}{\sigma} + \frac{\sum_{i=1}^{n}(x_i - \bar{x})^2 + n(\bar{x} - u)^2}{\sigma^3}
\end{aligned}
\tag{3.11}
$$

Which is solved by

$$\hat{\sigma}^2 = \sum_{i=1}^{n} (x_i - \hat{u}) / n \tag{3.12}$$

Inserting $\hat{u}$ we obtain

$$
\begin{aligned}
\hat{\sigma}^2 &= \frac{1}{n} \sum_{i=1}^{n} (x_i - \bar{x})^2 / n \\
\hat{\sigma}^2 &= \frac{1}{n} \sum_{i=1}^{n} x_i^{\,2} - \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} x_i x_j
\end{aligned}
\tag{3.13}
$$

To calculate its expected value, it is convenient to rewrite the expression in terms of zero-mean random variables $\delta_i \equiv u - x_i$. Expressing the estimate in these variables yields

$$\hat{\sigma}^2 = \frac{1}{n} \sum_{i=1}^{n} (x_i - \delta_i)^2 - \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} (u - \delta_i)(u - \delta_j) \tag{3.14}$$

Simplifying the expression above, utilizing the facts that $E[\delta_i] = 0$ and $E[\delta_i^2] = \sigma^2$, allows us to obtain

$$E[\hat{\sigma}_i^2] = \frac{n-1}{n}\sigma^2 \tag{3.15}$$

This means that the estimator $\hat{\sigma}$ is biased. However, $\hat{\sigma}$ is consistent.

Formally we say that the maximum likelihood estimator for $\theta = (\mu, \sigma)^2$ is: $\hat{\theta} = (\hat{\mu}, \hat{\sigma})^2$

3.3 Motion in two dimensions

The position is represented by the vector **r**. The velocity will still be represented by **v** and the acceleration by **a**. In general, the average velocity will be given by:

$$\bar{v} = \Delta r / \Delta t \tag{3.16}$$

The instantaneous velocity is given by a similar formula, with the condition that a very small time interval is used to measure the displacement.

A similar formula gives the average acceleration: $\bar{a} = \Delta v / \Delta t$

Again, the instantaneous acceleration is found by measuring the change in velocity over a small time interval.

3.4 RGB color model

The RGB color model is an additive color model in which red, green, and blue light are added together in various ways to reproduce a broad array of colors. The name of the model comes from the initials of the three additive primary colors, red, green, and blue. The main purpose of the RGB color model is for the sensing, representation, and display of images in electronic systems, such as televisions and computers, though it has also

been used in conventional photography. Before the electronic age, the RGB color model already had a solid theory behind it, based in human perception of colors.

RGB is a device-dependent color model: different devices detect or reproduce a given RGB value differently, since the color elements and their response to the individual R, G, and B levels vary from manufacturer to manufacturer, or even in the same device over time. Thus an RGB value does not define the same color across devices without some kind of color management.

Typical RGB input devices are color TV and video cameras, image scanners, and digital cameras. Typical RGB output devices are TV sets of various technologies computer and mobile phone displays, video projectors, multicolor LED displays.

3.5 Sampling With Replacement

Suppose we have 100 particles numbers from 1 to 100. We want to select a random sample of numbers of particle. After we pick a number of particle, we can put the number aside or we can put it back into the group of particles. If we put the number back in the group of particles, it may be selected more than once. When a population element can be selected more than one time, we are sampling with replacement

3.6 Air Bearing Surface

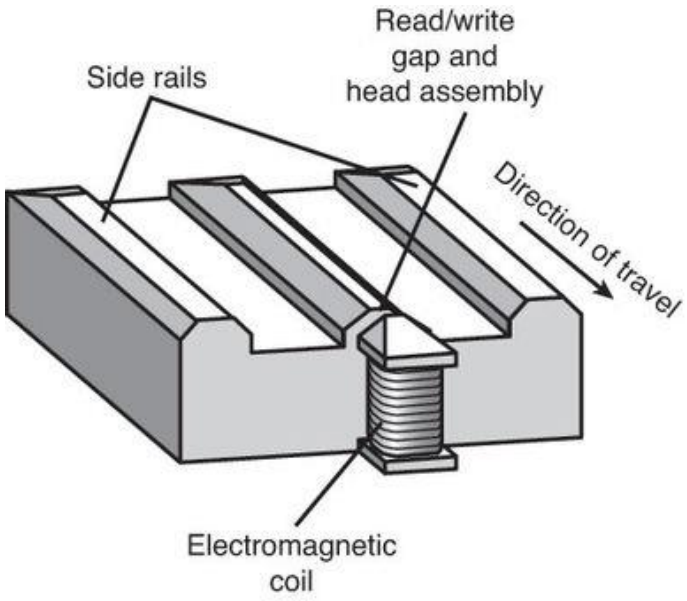


Figure 3.2 ABS

Air will passing under the air bearing surfaces (ABS) for lifting slider over media surface. ABS design was concerned by (has to take into account) the weight, velocity, and skew to achieve a uniformed flying height. Fig. 1 shows the typical ABS structure. In the slider attachment process, the position of ABS needs to be inspected before moving the slider and putting on the suspension for attachment.

CHAPTER 4
PARTICLE FILTER DETECT SLIDER

This research purposes the alternative ways to come over the unit per hour issue of the currently in used vision system, need to stop robot before capture the picture. The Particle filter detection algorithm is developed to detect the moving slider on the screen of vision by using particle filter and resampling according to likelihood function technique Moreover, this feasibility will be help to increase unit per hour of the machine and we will find the parameter that can detect moving slider faster.

We can predict the moving slider trajectory by solving differential equation
The problem of solving differential equation are

-Simulation of long period of time might cause accumulation of error
-Smallest error of initial value might cause a drastic change of solution

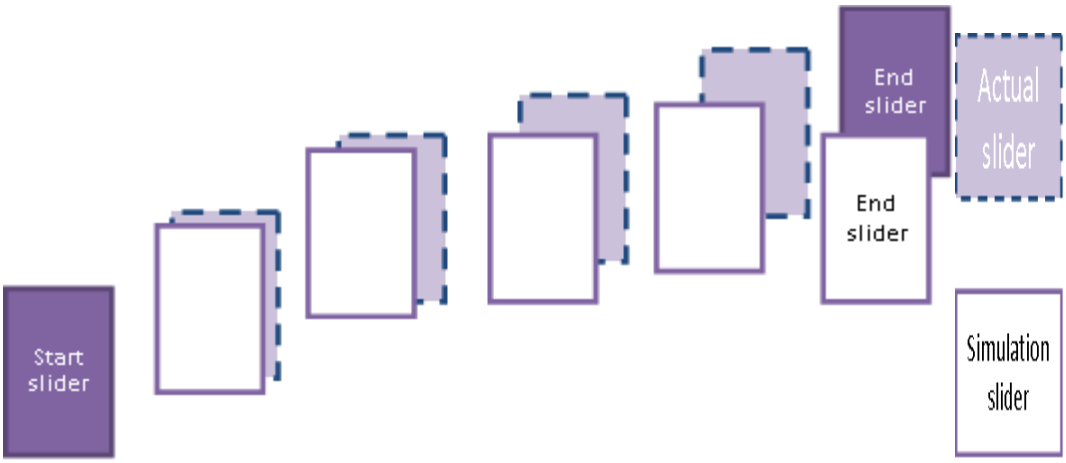


Figure 4.1 the accumulation of error

We can use the good measurement to solve the error's problem

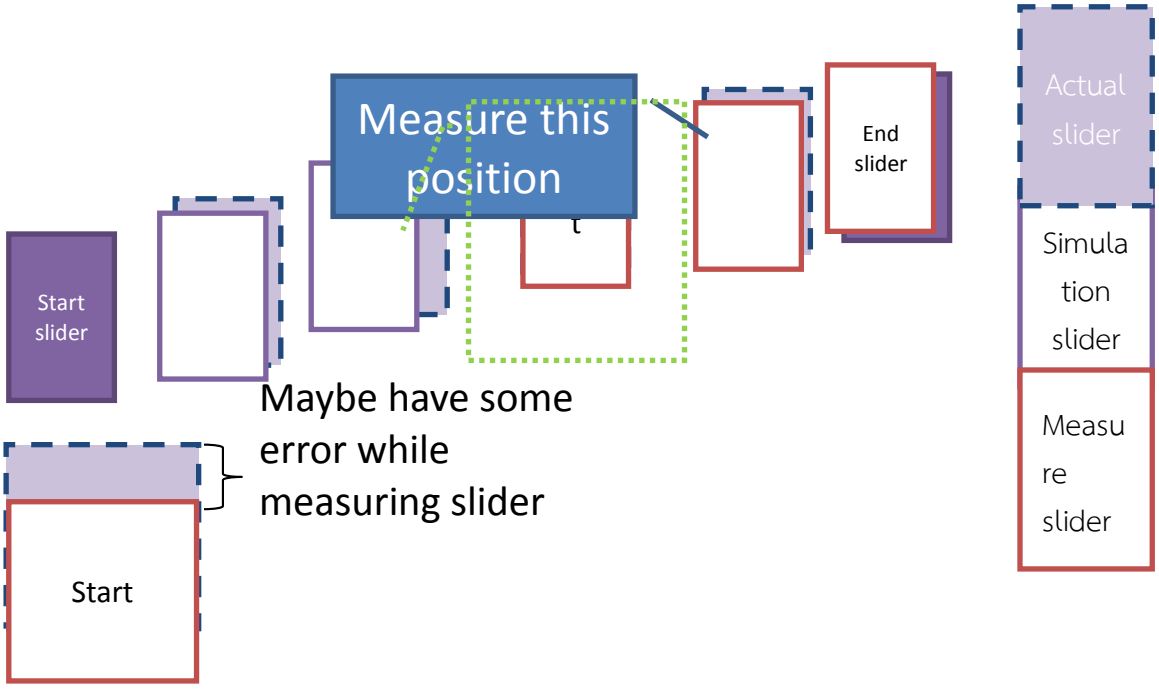


Figure 4.2 the error while measuring slider

Combining simulation and measured data is the way to get assimilating data by particle filter

4.1 State space model

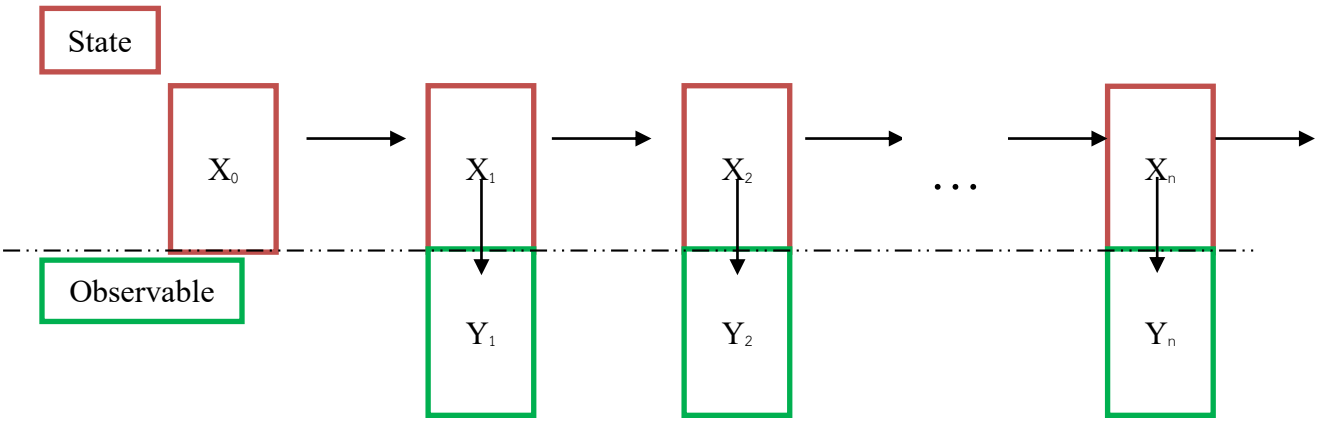


Figure 4.3 State space model

System model : $$x_n = f(x_{n-1}) + W_n) \quad (4.1)$$

Observation model : $$Y_n = G(Y_{n-1}) + V_n) \quad (4.2)$$

W_n denote the system noise ;

V_n denote the observation noise ;

We generate sample from system model which follow filter distribution

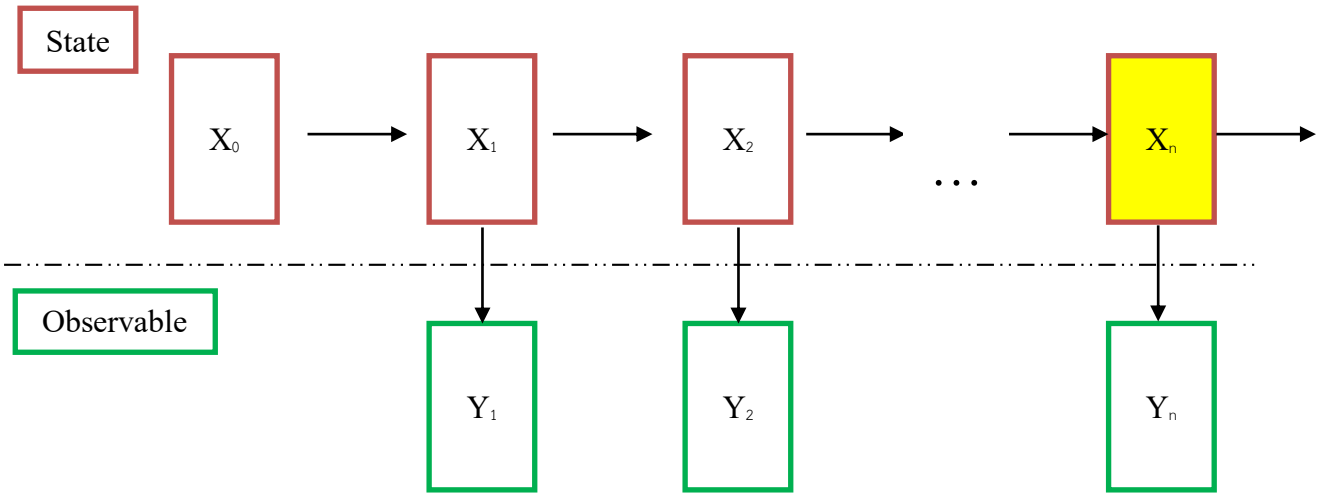


Figure 4.4 Generate state form filter distribution

$$P(X_n \mid Y_{1:n}) = \int\!\!\cdots\!\!\int P(X_{0:n} \mid Y_{1:n}) dX_{n-1} \ldots dX_0 \quad (4.3)$$

We estimate the state by mean or median method

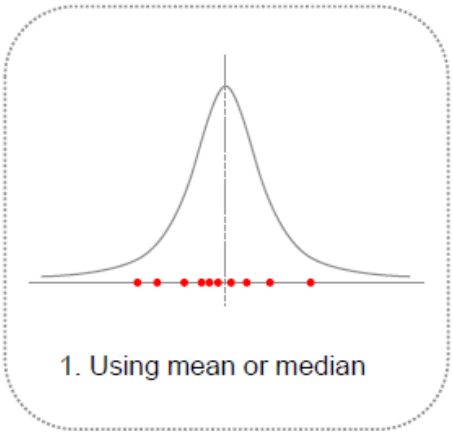


Figure 4.5 Mean or median method

4.2 Predict the state of particle

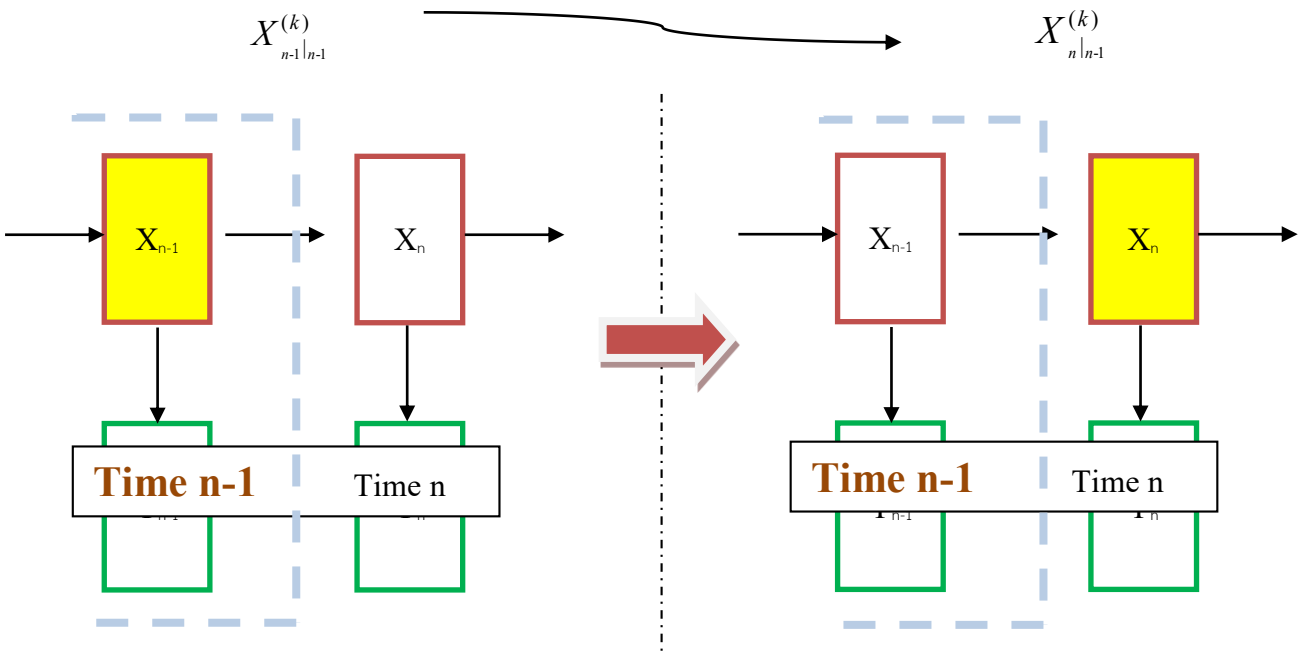


Figure 4.6 Predict the state of particle

$$X_{n-1|n-1}^{(k)} \Rightarrow X_n f(X_{n-1|n-1}^{(k)}) + W_n$$

We predict particle state base on 2D movement State location and speed of slider

$$X = x, y, x', y'$$

$$\begin{pmatrix} x_n \\ y_n \\ x'_n \\ y'_n \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_{n-1} \\ y_{n-1} \\ x'_{n-1} \\ y'_{n-1} \end{pmatrix} + \begin{pmatrix} W_x \\ W_y \\ W_{x'} \\ W_{y'} \end{pmatrix} \quad (4.4)$$

4.3 Filtering the particle

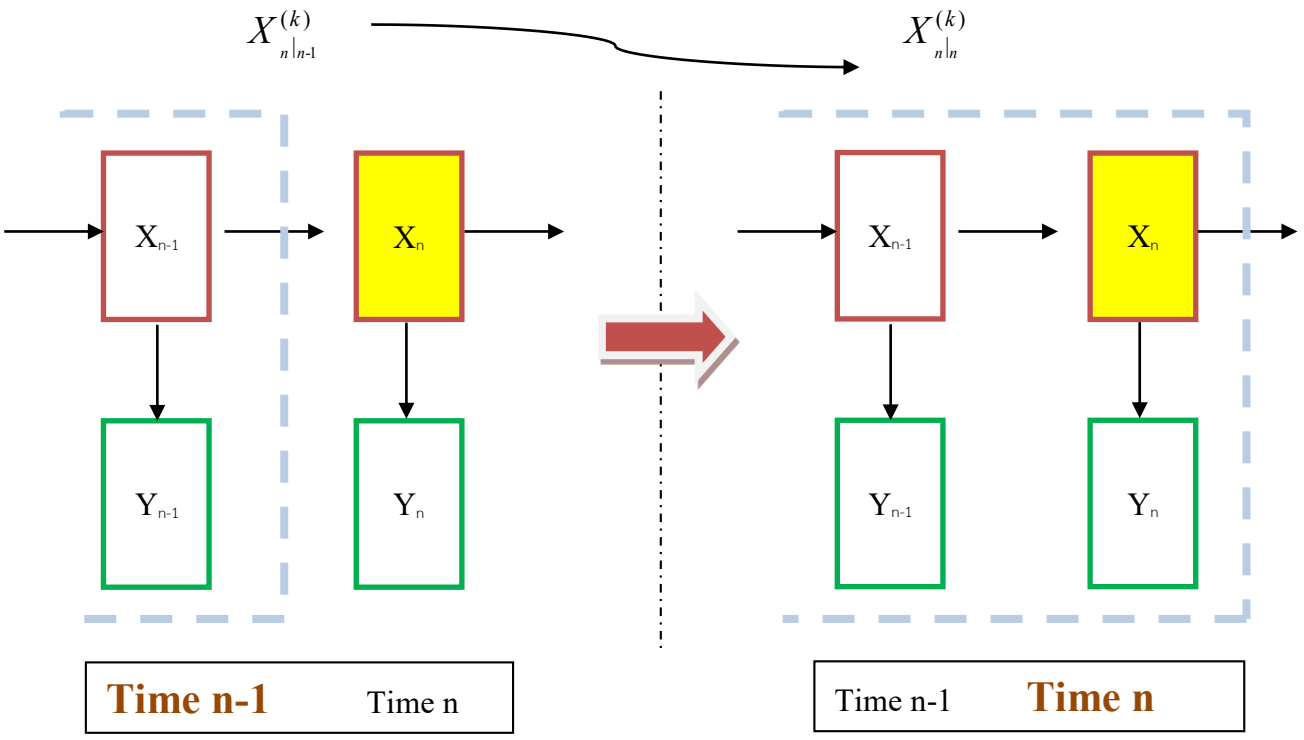


Figure 4.7 Observe particle state base on time n

$X_{n|n}^{(k)}$ Estimator at time n using information up to time n

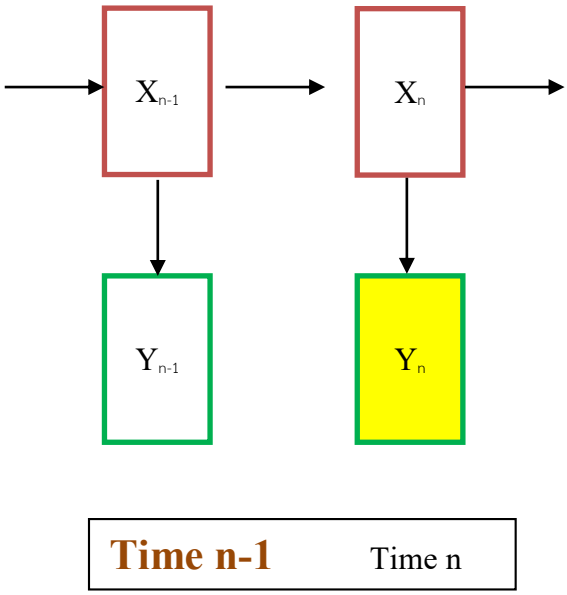


Figure 4.8 Measure color of pixel on particle Observable Y_n when state is $X_{n|n-1}^{(k)}$

After we estimate X_n(the Location of particle (x, y) we will get the color of pixel on which particle exists in R, G, B color then compare the R, G, B color of pixel with target color (slider is white color 255, 255, 255)

4.3.1 Likelihood

We are supposing normal distribution for simplicity

$$P(Y_n \mid X_{n|n-1}^{(i)}) = \frac{1}{\sqrt{2\pi\sigma}} \exp(-\frac{d^2}{2\pi\sigma^2}) \quad (4.5)$$

When $d = \sqrt{(R - r)^2 + (G - g)^2 + (B - b)^2}$

R, G, B denote the pixel's colors.

r, g, b denote the target colors.

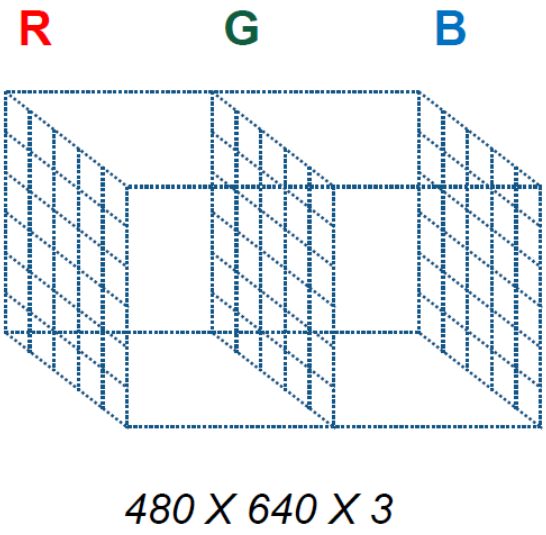


Figure 4.9 R G B color in color photo

4.3.2 Resampling

We use resampling with replacement according to likelihood ratio

$$\frac{P(Y_n \mid X_{n|n-1}^{(i)})}{\sum_{-k}^{k} P(Y_n \mid X_{n|n-1}^{(i)})} - (k = 1,\ldots,N) \tag{4.6}$$

Particles with high likelihood are more likely to be picked up

4.4 Program algorithm

After we generate particle base on state space model the program will predict the next state of particle and reselection particles according to their likelihood in the filtering function the program will do the loop until the object out of screen

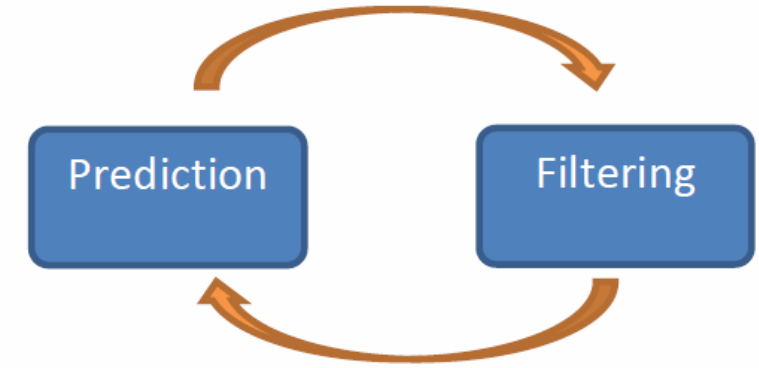


Figure 4.10 Program algorithm

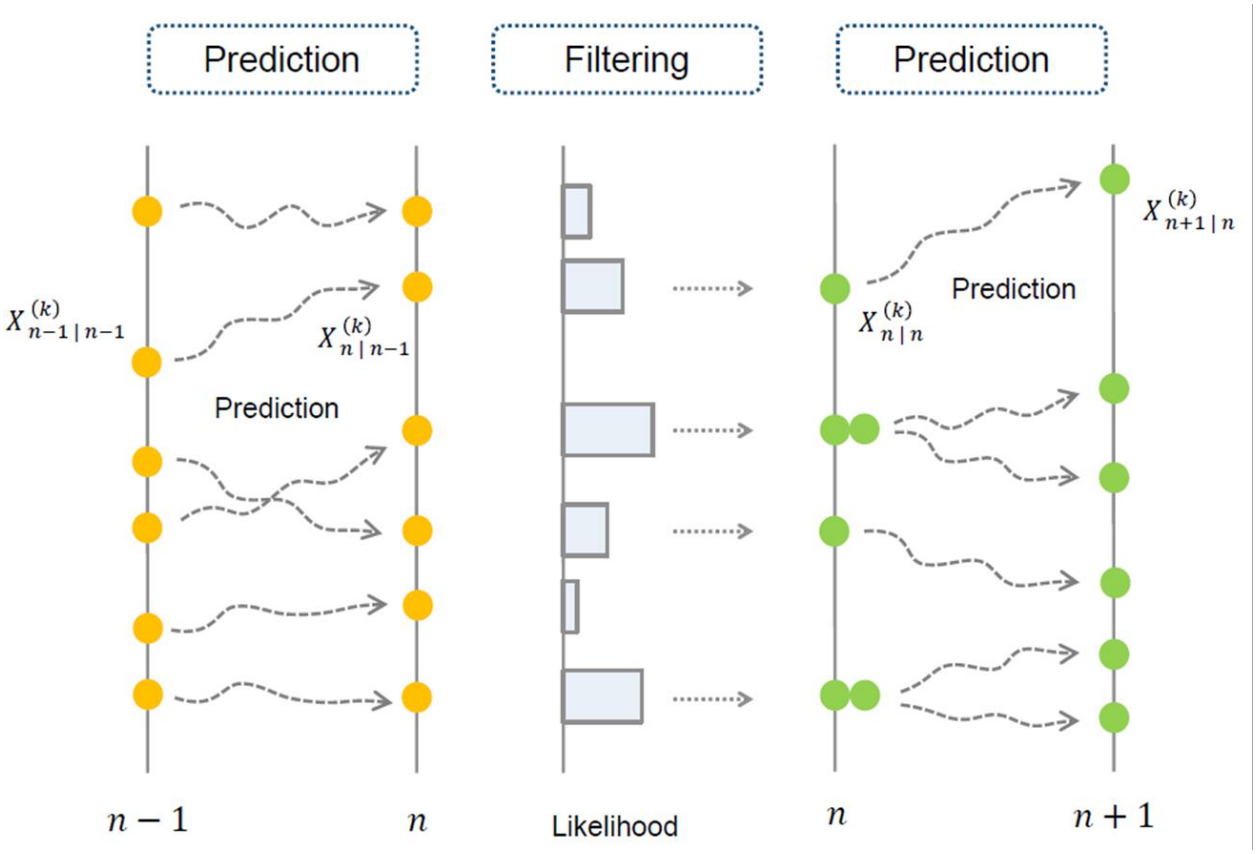


Figure 4.11 How particle move

CHAPTER 5
PARTICLE FILTER DETECT SLIDER

5.1 Result and Discussion

In this research work, particle filter is applied to inspect the ABS on the slider at the slider attachment machine. To verify the effectiveness of the proposed algorithm, the performance in terms of the number of the object detection frame was investigated. In addition, noise of the movement is added to the system for verifying the robustness of the proposed algorithms. In the experiments, the test by adjusting the parameters standard deviation of RGB color σ, the particle number and noise of the movement metric were performed. In the video stream from the vision system, the slider has completely been appeared on screen at the 44^{th} of the video frame (screen pixel 600*800, video frame is 390 frame, the light at the center of the video is flare). The followings are the parameters attempted to be adjusted in the experiments.

- Standard deviation σ = 20, 50, 100, 200
- Number of particles N = 100, 1000
- Noise of the movement level values (E) = 25, 50, 100
- Number of repeated experiments = 4

Table 5.1 The results of the frame number that first detect the object correctly when the number of particles is 100 and noise of movement level is 25.

	Standard deviation σ	20	50	100	200
The first frame that particle can find the object	Test No.1(frame order)	69^{th}	68^{th}	66^{th}	54^{th}
	Test No.2(frame order)	67^{th}	67^{th}	69^{th}	50^{th}
	Test No.3(frame order)	70^{th}	67^{th}	68^{th}	52^{nd}
	Test No.4(frame order)	69^{th}	69^{th}	69^{th}	58^{th}
	Average(frame order)	68^{th}	67^{th}	68^{th}	53^{rd}

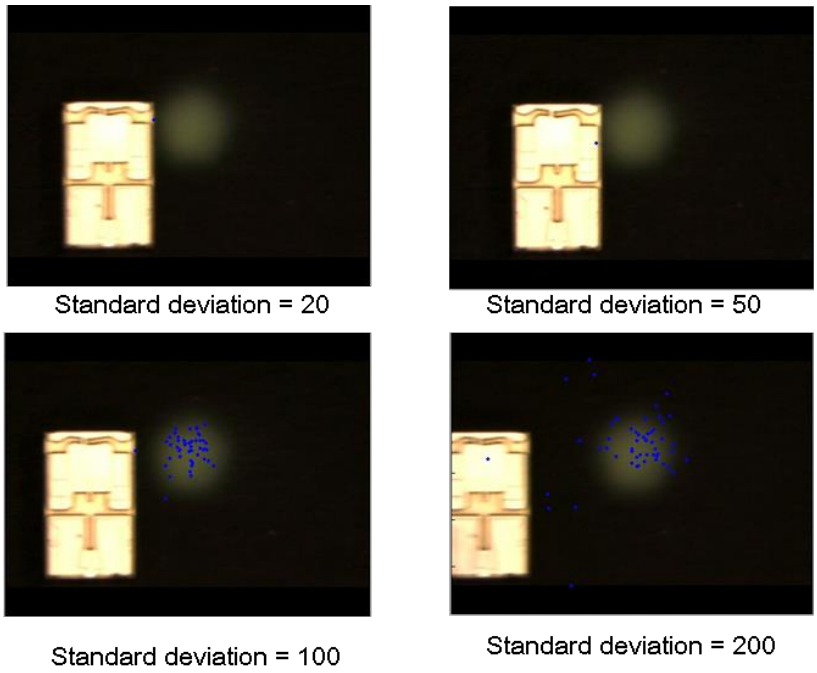


Figure 5.1 The first frame that the particle finds the slider : N = 100 , noise level = 25

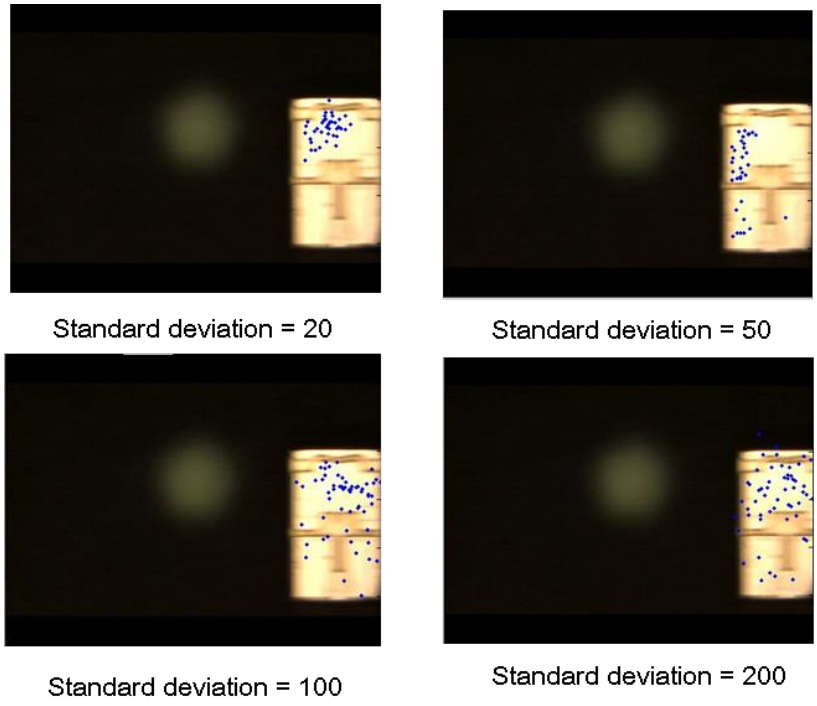


Figure 5.2 Resampling after the particles find the slider in the screen: N = 100, noise level = 25

Table 5.2 The results of the first frame number that detect the object correctly when the number of particles is 1000 and noise of movement level is 25.

	Standard deviation σ	20	50	100	200
The first frame that particle can find the object	Test No.1(frame order)	66th	66th	62nd	51st
	Test No.2(frame order)	67th	66th	63rd	50th
	Test No.3(frame order)	65th	65th	62nd	53rd
	Test No.4(frame order)	67th	66th	61st	52nd
	Average(frame order)	66th	65th	62nd	51st

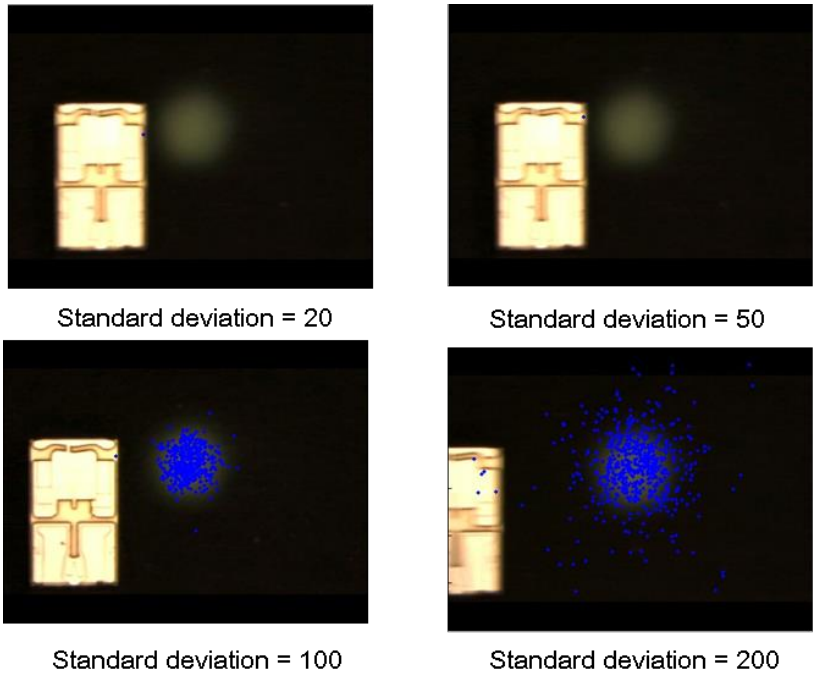


Figure 5.3 The first frame that the particle finds the slider: N = 1000 , noise level = 25

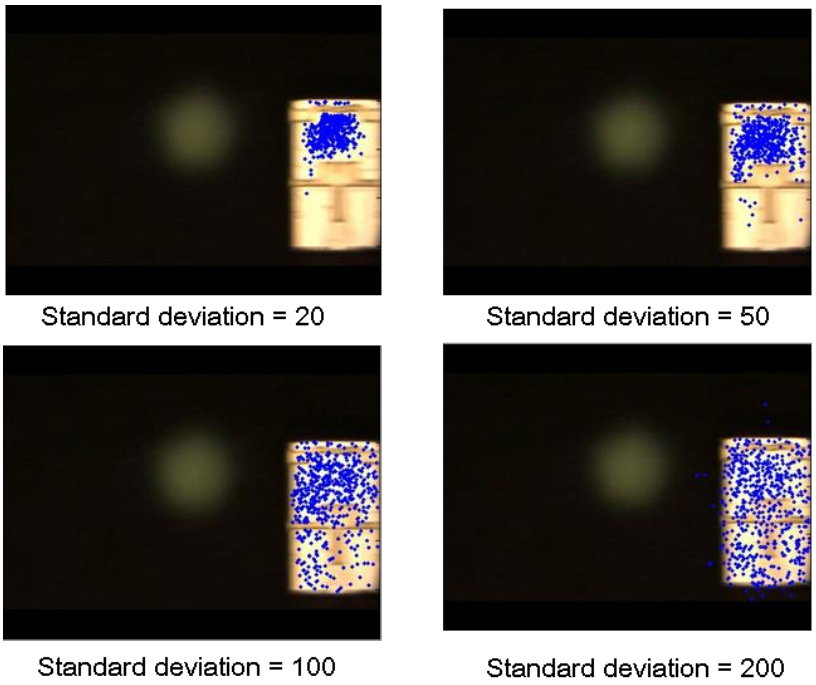


Figure 5.4 Resampling after the particles find the slider in the screen: N = 1000, noise level = 25

Table 5.3 The results of the first frame number that detect the object correctly when the number of particles is 1000 and noise of movement level is 50.

	Standard deviation σ	20	50	100	200
The first frame that particle can find the object	Test No.1(frame order)	51st	58th	58th	44th
	Test No.2(frame order)	54th	61st	57th	45th
	Test No.3(frame order)	55th	66th	57th	44th
	Test No.4(frame order)	54th	62nd	56t	46th
	Average(frame order)	53rd	61st	57th	45th

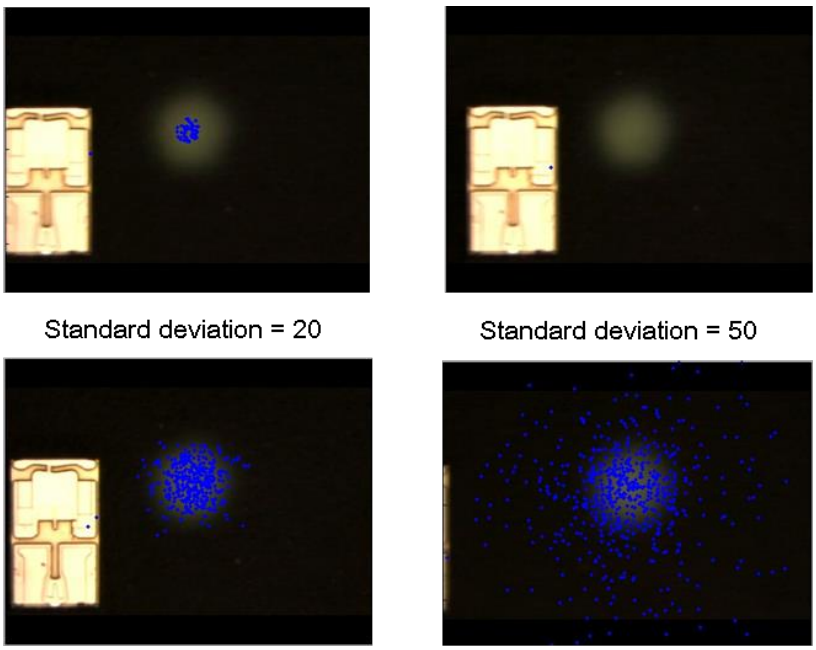


Figure 5.5 The first frame that the particle finds the slider: N = 1000 , noise level = 50

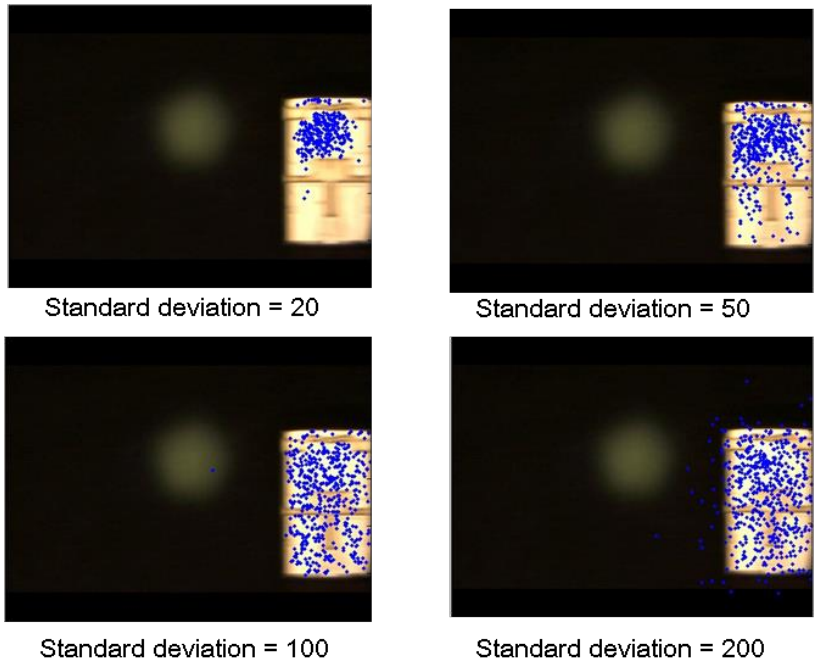


Figure 5.6 Resampling after the particles find the slider in the screen: N = 1000, noise level = 50

Table 5.4 The results of the first frame number that detect the object correctly when the number of particles is 1000 and noise of the movement level is 100.

	Standard deviation σ	20	50	100	200
The first frame that particle can find the object	Test No.1(frame order)	50^{th}	51^{st}	46^{th}	44^{th}
	Test No.2(frame order)	49^{th}	48^{th}	47^{th}	44^{th}
	Test No.3(frame order)	50^{th}	46^{th}	47^{th}	44^{th}
	Test No.4(frame order)	50^{th}	50^{th}	46^{th}	45^{th}
	Average(frame order)	49^{th}	48^{th}	46^{th}	44^{th}

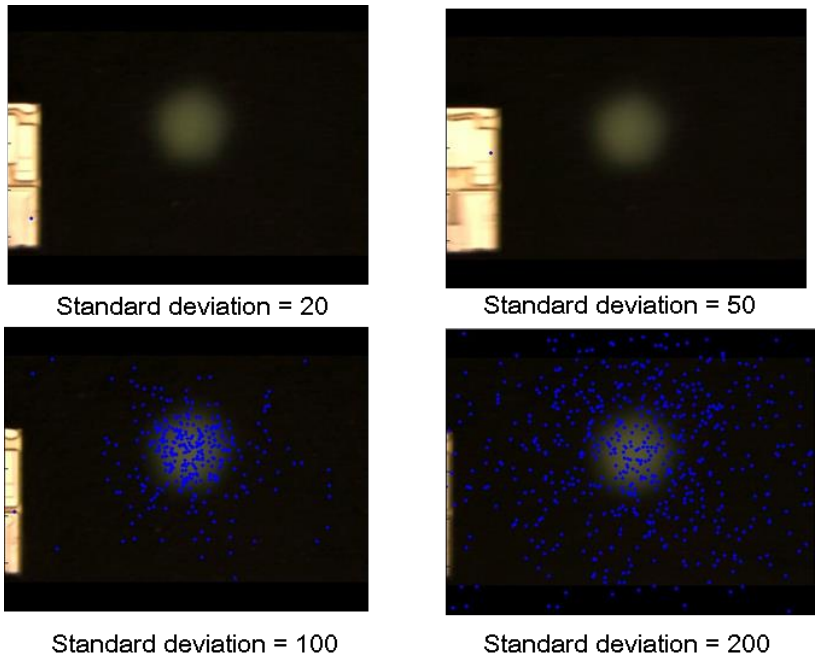


Figure 5.7 The first frame that the particle finds the slider: N = 1000 , noise level = 100

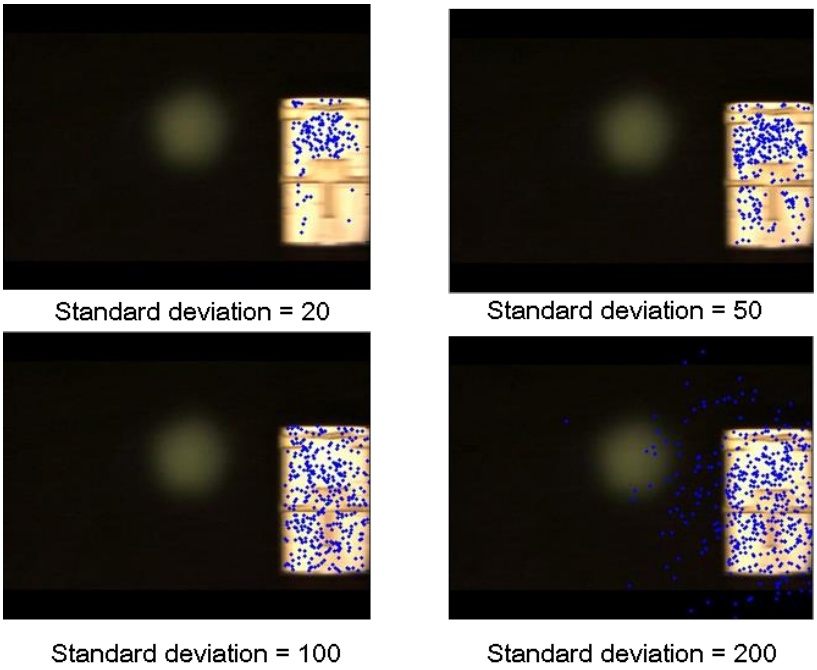


Figure 5.8 Resampling after the particles find the slider in the screen: N = 1000, noise level = 100

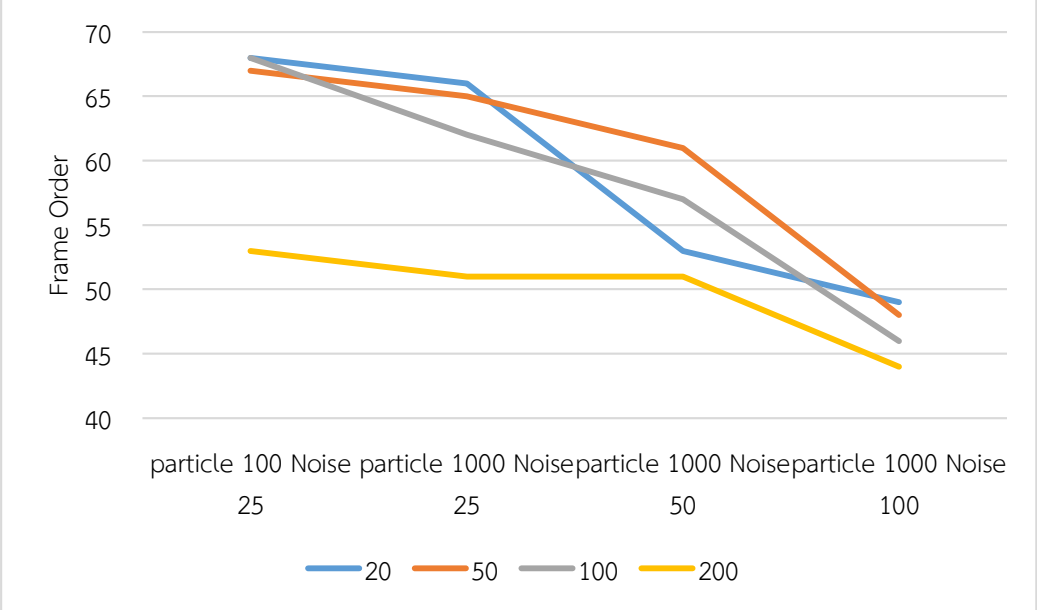


Figure 5.9 Line Chart Compare frame order that can detect slider between each parameters

The result from table shown that the particle detect flare at the 1st time and then more particles, high noise level values and more standard deviation can detect slider

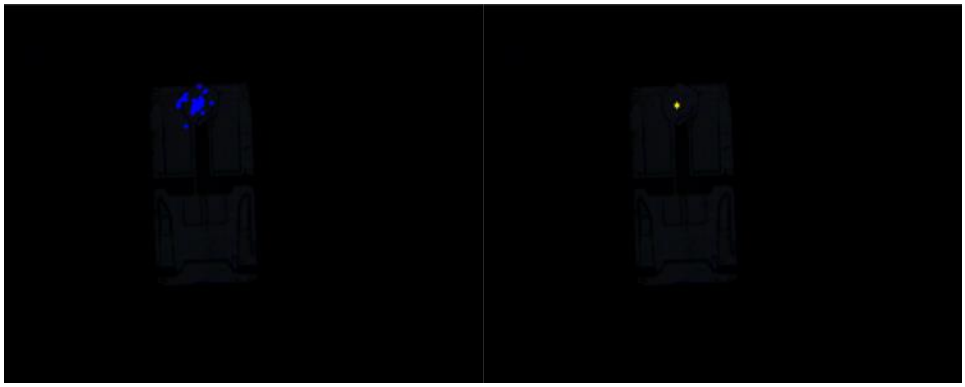
faster but the particles number direct variation with and particles of parameter standard deviation (200) cannot find the slider accurately so we need to optimize the parameter.

Next experiments, test by adjusting the parameters standard deviation and noise of the movement metric were performed. In the video stream from the vision system, the slider is on the screen and the light is increased from 0 – 100 % (screen pixel 600*800 ,slider area is 320*176 pixels, frame number 0 – 390, center of the slider is on pixel (275,337)), is finding the proper parameter that robust to the light environment. The followings are the parameters attempted to be adjusted in the experiments.

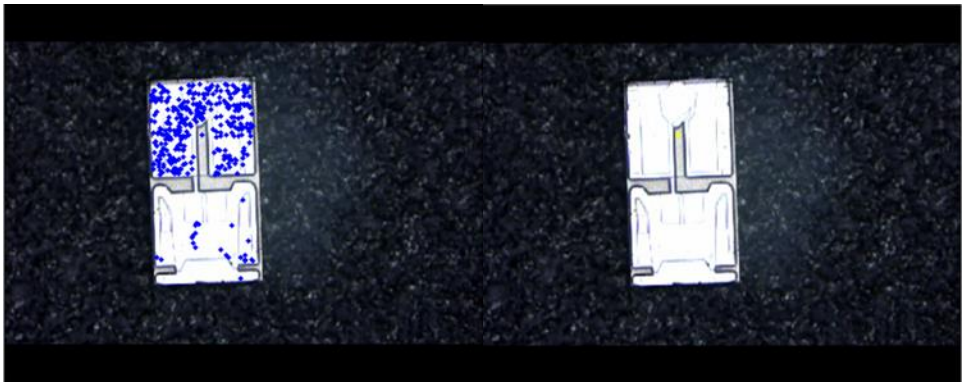
- Standard deviation σ = 20, 50, 100, 200
- Noise level values (E) = 0 (basic particles filter), 25, 50, 100
- Number of repeated experiments = 4

Table 5.5 The results of the frame number that first detect and the last frame that can detect the object correctly when the standard deviation is 20.

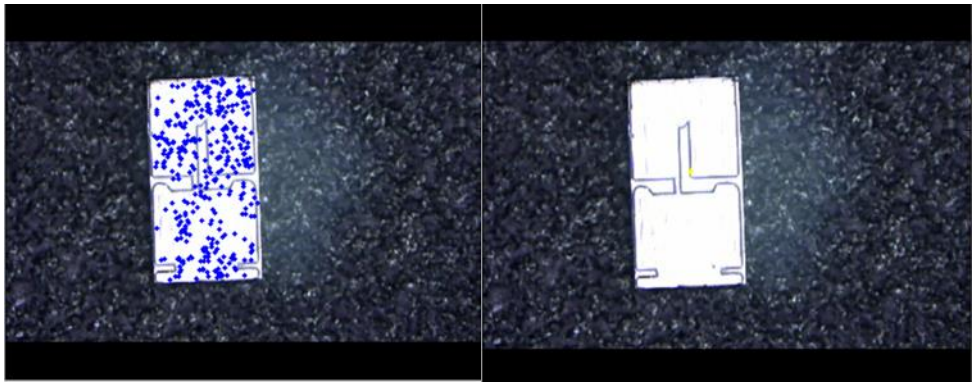
Noise movement level values	0		25	
Frame order	Frame Detect	Last Frame detect	Frame Detect	Last Frame detect
Test No.1	47th	331st	47th	331st
Test No.2	46th	332nd	47th	332nd
Test No.3	47th	332nd	48th	331st
Test No.4	47th	333rd	47th	332nd
Average	47th	332nd	47th	332nd
Noise movement level values	50		100	
Frame order	Frame Detect	Last Frame detect	Frame Detect	Last Frame detect
Test No.1	48th	330th	50th	322nd
Test No.2	47th	329th	50th	324th
Test No.3	47th	330th	50th	324th
Test No.4	48th	330th	50th	324th
Average	47th	330th	50th	324th



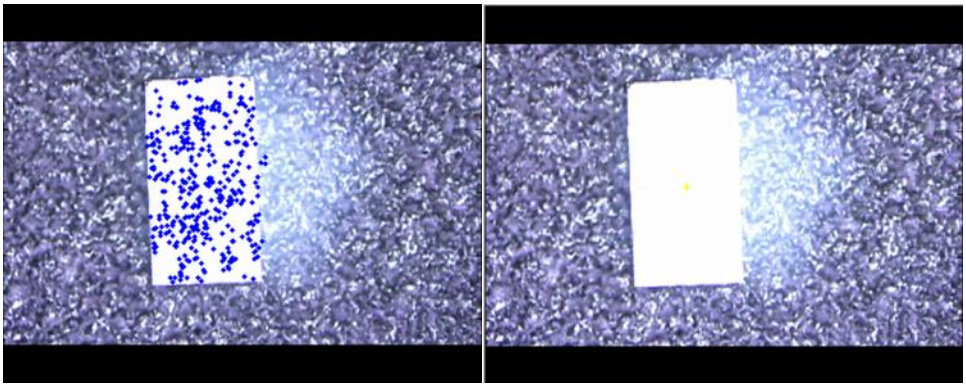
(A) (B)



(C) (D)



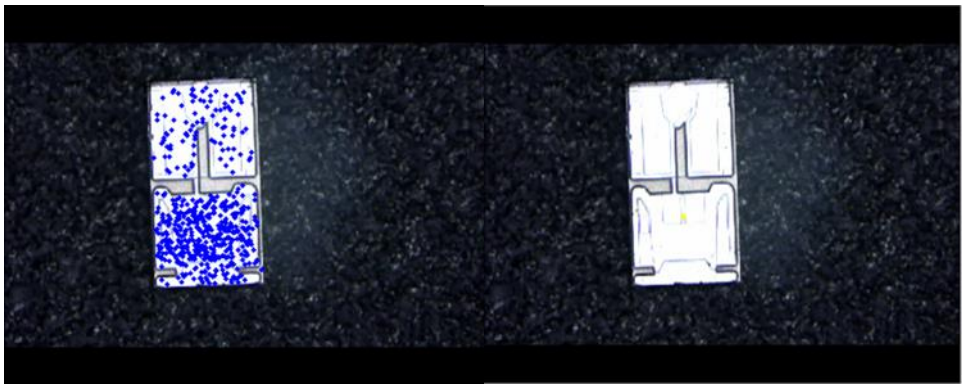
(E) (F)



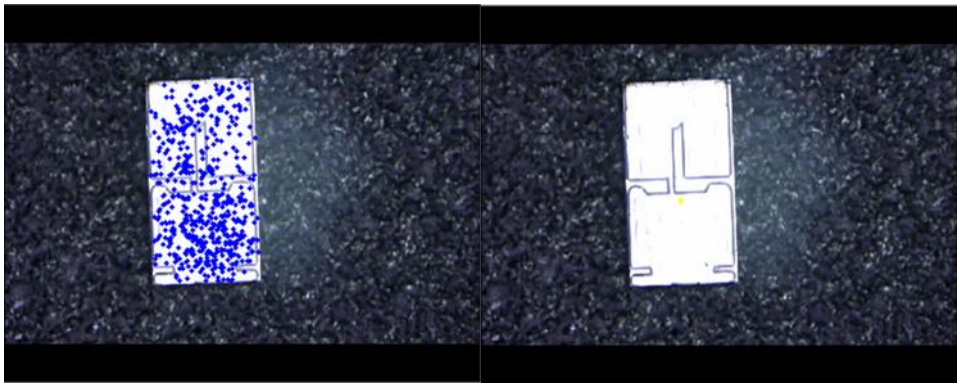
(G) (H)

Figure 5.10 (A) 1st time particle can detect slider correctly 47th frame
(B), (D), (F), (H) Mean of particles 47th, 90th, 150th, 332nd frame
(C), (E), (G) particle can detect slider correctly 90th, 150th, 332nd frame
(standard deviation 20 noise level values 0)

(A) (B)



(C) (D)

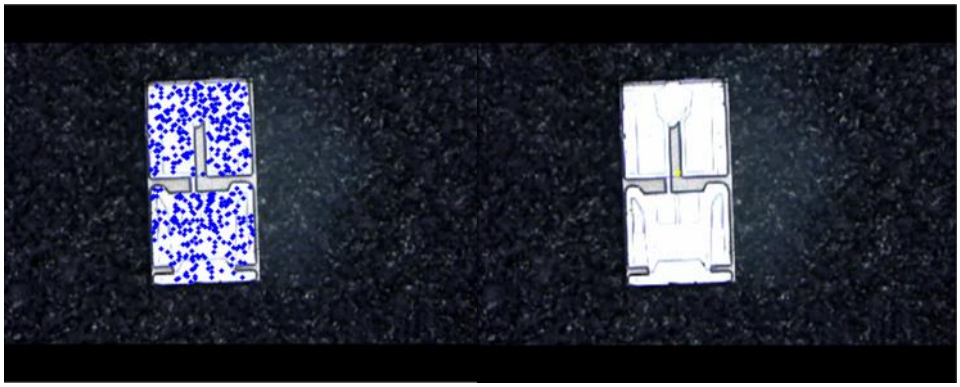


(E) (F)

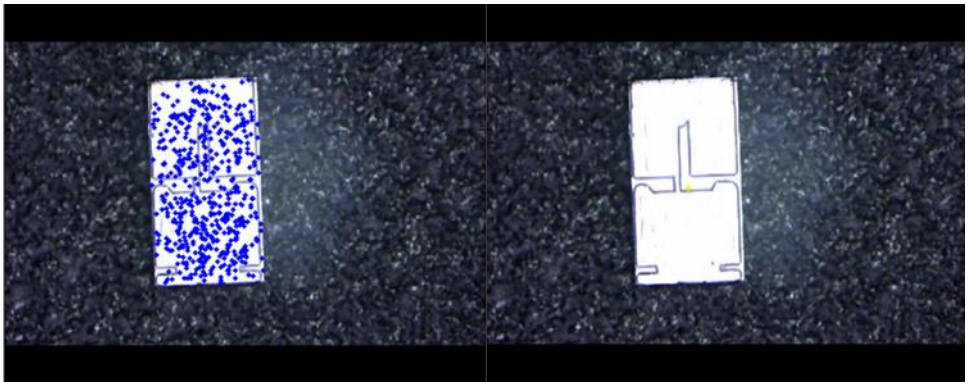
(G) (H)

Figure 5.11 (A) 1st time particle can detect slider correctly 47th frame
(B), (D), (F), (H) Mean of particles 47th, 90th, 150th, 332nd frame
(C), (E), (G) particle can detect slider correctly 90th, 150t, 332nd frame
(standard deviation 20 noise level values 25)

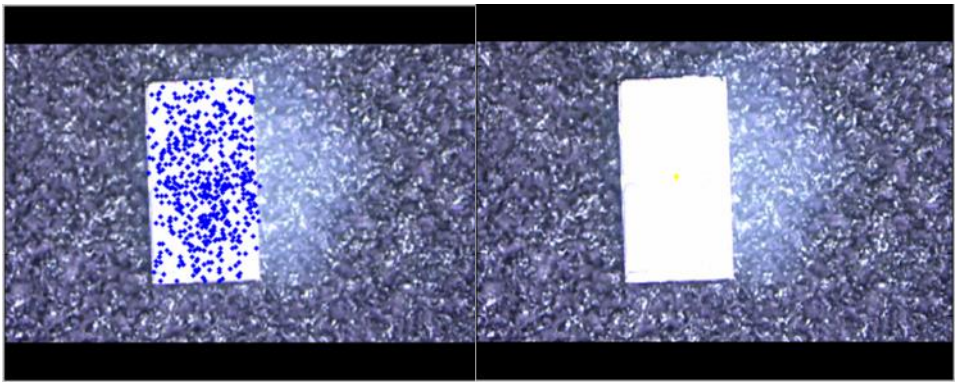
(A) (B)



(C) (D)



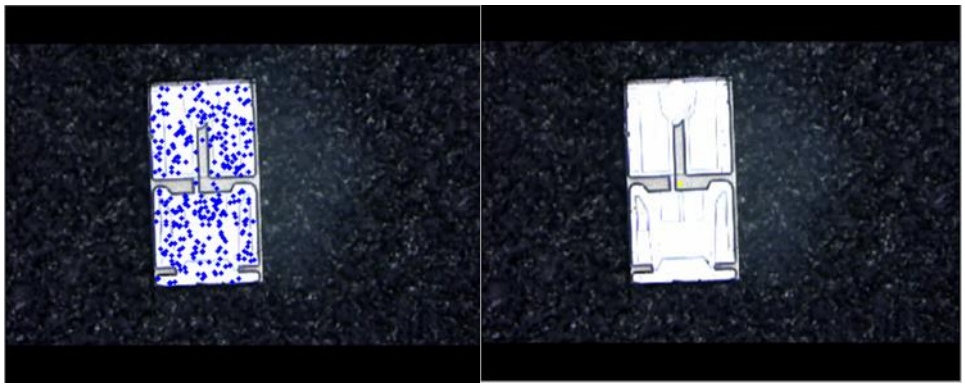
(E) (F)



(G) (H)

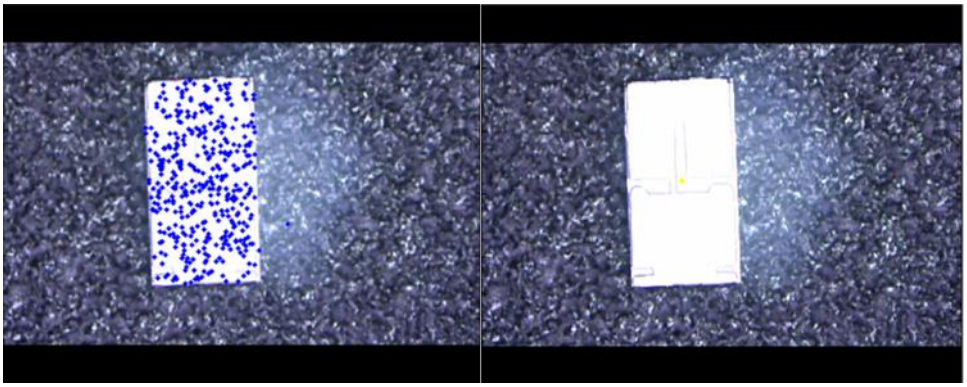
Figure 5.12 (A) 1st time particle can detect slider correctly 47th frame
(B), (D), (F), (H) Mean of particles 47th, 90th, 150th, 330th frame
(C), (E), (G) particle can detect slider correctly 90th, 150th, 330th frame
(standard deviation 20 noise level values 50)

(A) (B)



(C) (D)

(E) (F)



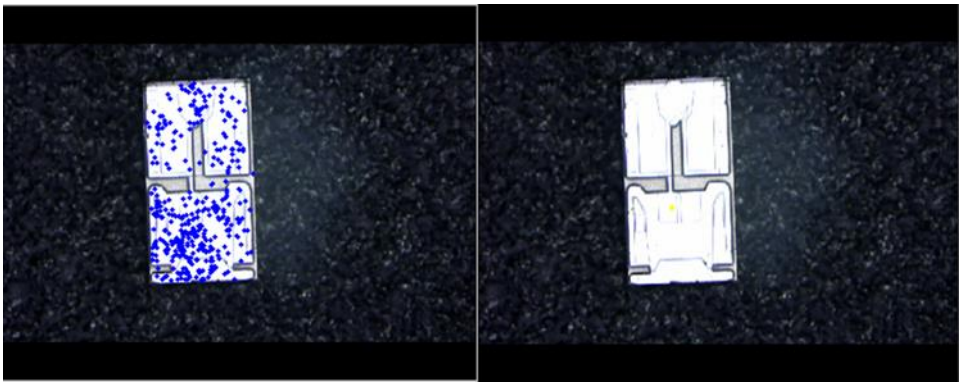
(G) (H)

Figure 5.13 (A) 1st time particle can detect slider correctly 50th frame (B), (D), (F), (H) Mean of particles 50th, 90th, 150th , 324th frame (C), (E), (G) particle can detect slider correctly 90th, 150th, 324th frame (standard deviation 20 noise level values 100)

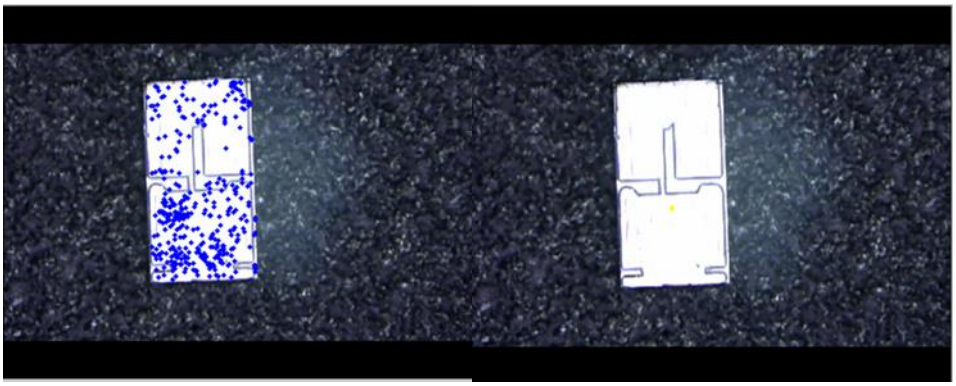
Table 5.6 The results of the frame number that first detect and the last frame that can detect the object correctly when the standard deviation is 50.

Noise movement level values	0		25	
Frame order	Frame Detect	Last Frame detect	Frame Detect	Last Frame detect
Test No.1	50th	326th	53rd	326th
Test No.2	50th	328th	53rd	327th
Test No.3	51st	328th	54th	326th
Test No.4	50th	328th	53rd	326th
Average	50th	328th	53rd	326th
Noise movement level values	50		100	
Frame order	Frame Detect	Last Frame detect	Frame Detect	Last Frame detect
Test No.1	55th	323rd	55th	288th
Test No.2	55th	323rd	55th	287th
Test No.3	56th	323rd	56th	289th
Test No.4	54th	324th	55th	288th
Average	55th	323rd	55th	288th

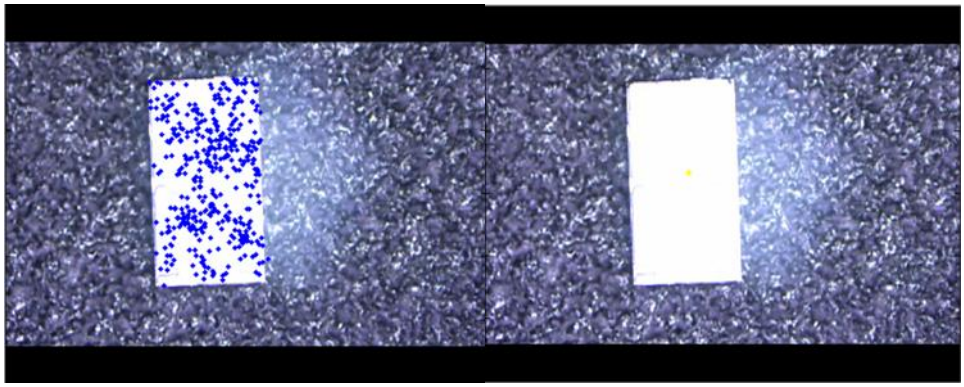
(A) (B)



(C) (D)



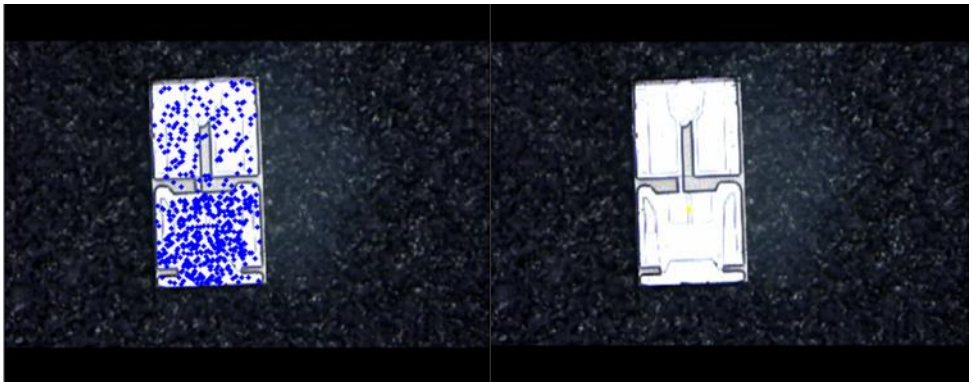
(E) (F)



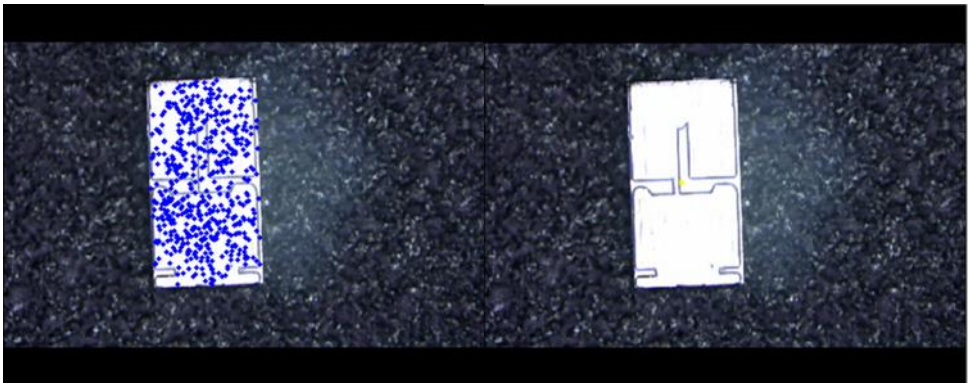
(G) (H)

Figure 5.14 (A) 1st time particle can detect slider correctly 50th frame (B), (D), (F), (H) Mean of particles 50th, 90th, 150th, 328nd frame (C), (E), (G) particle can detect slider correctly 90th, 150th, 328nd frame (standard deviation 50 noise level values 0)

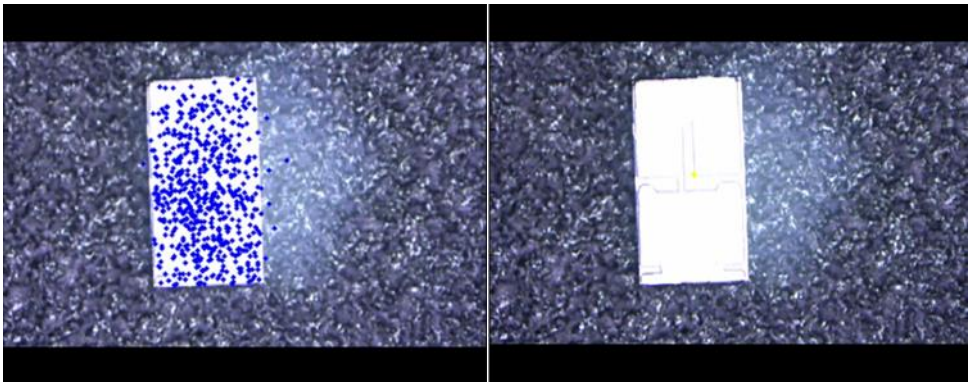
(A) (B)



(C) (D)

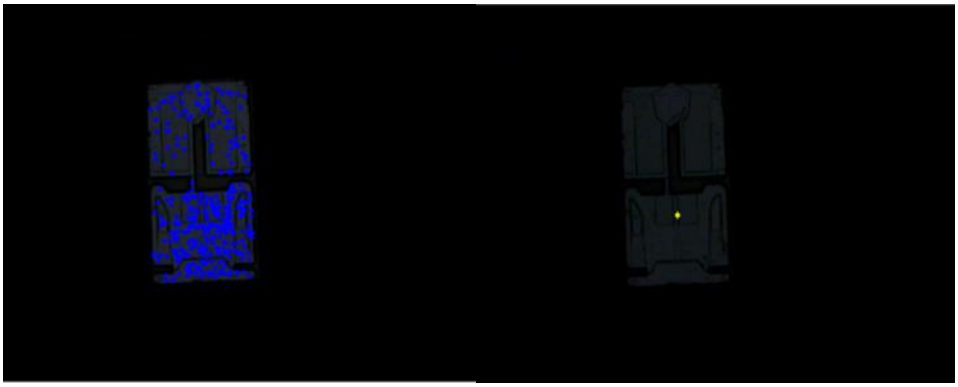


(E) (F)

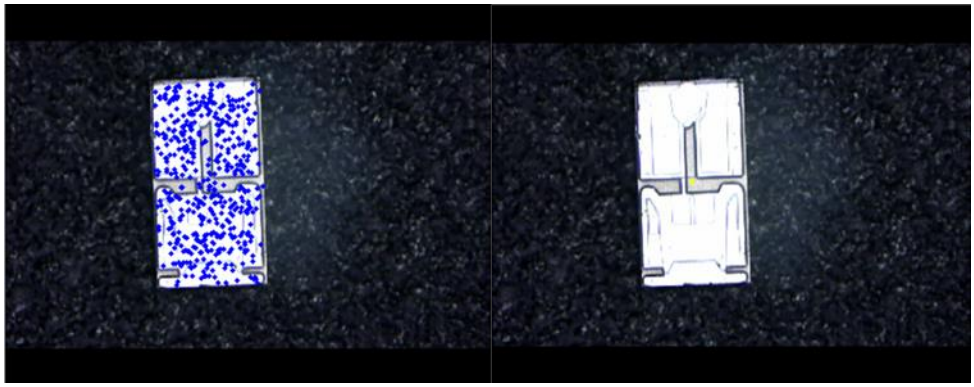


(G) (H)

Figure 5.15 (A) 1st time particle can detect slider correctly 53rd frame
(B), (D), (F), (H) Mean of particles 53rd, 90th, 150th, 326th frame
(C), (E), (G) particle can detect slider correctly 90th, 150th, 326th frame
(standard deviation 50 noise level values 25)

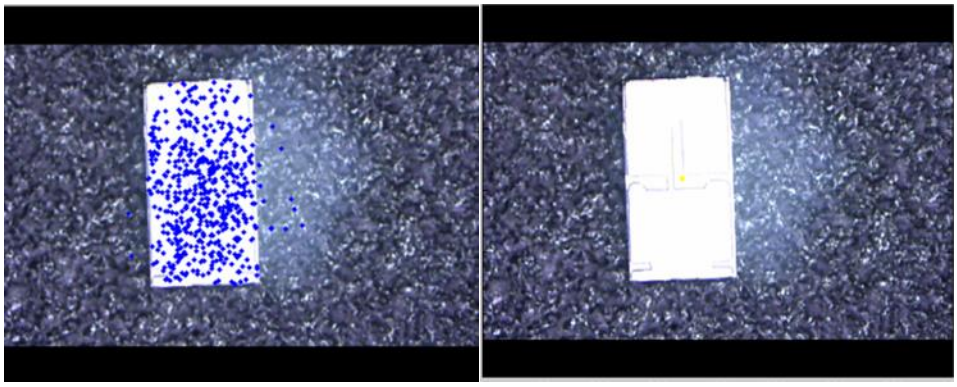


(A) (B)



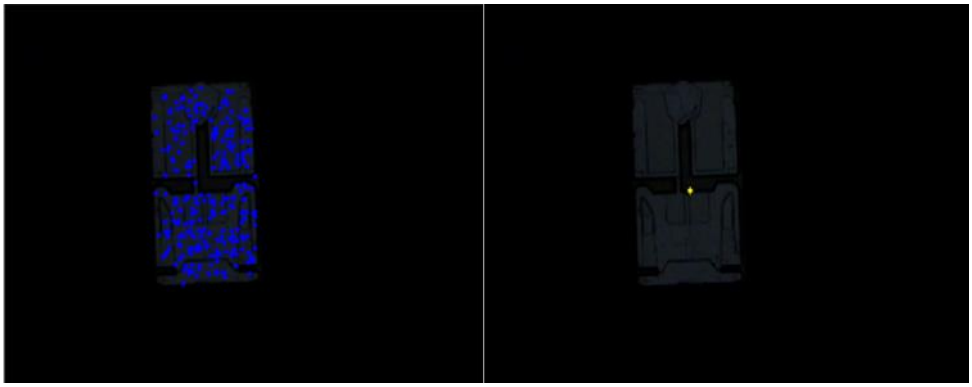
(C) (D)

(E) (F)

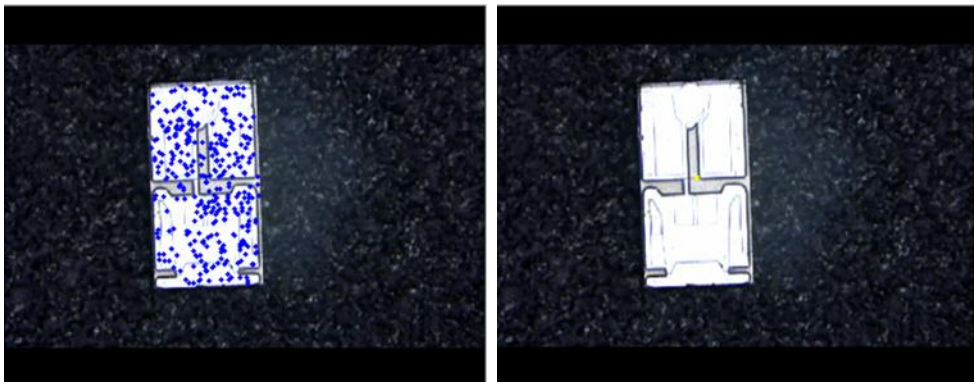


(G) (H)

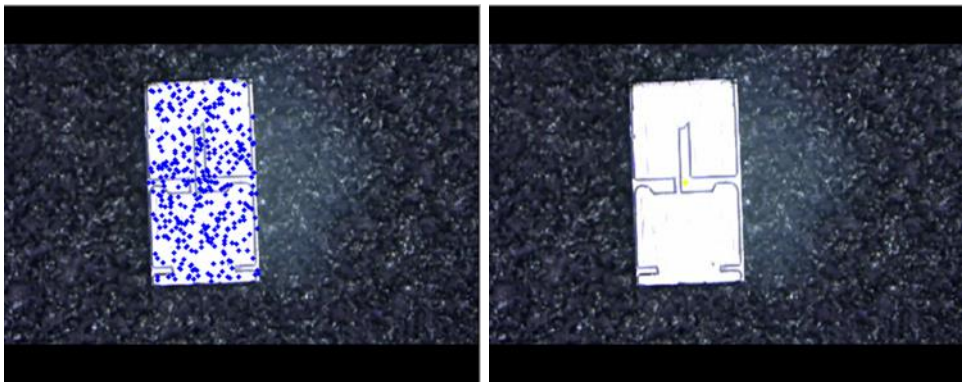
Figure 5.16 (A) 1st time particle can detect slider correctly 55th frame
(B), (D), (F), (H) Mean of particles 55th, 90th, 150th, 323rd frame
(C), (E), (G) particle can detect slider correctly 90th, 150th, 323rd frame
(standard deviation 50 noise level values 50)



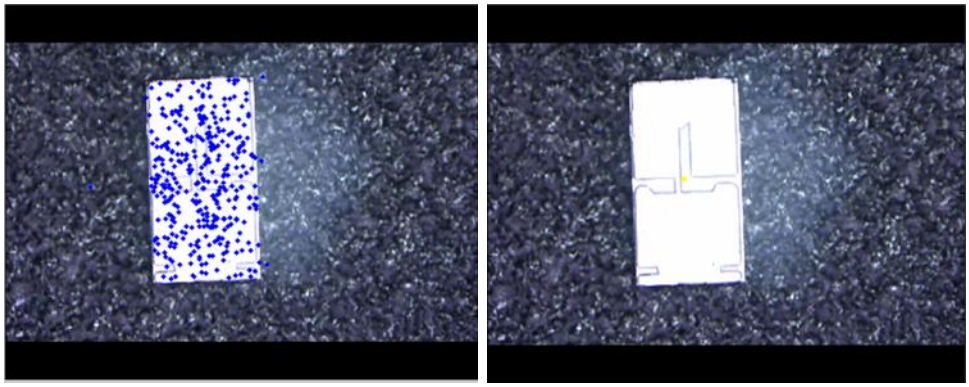
(A) (B)



(C) (D)



(E) (F)

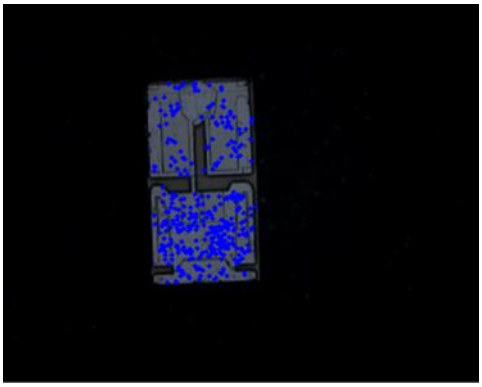


(G) (H)

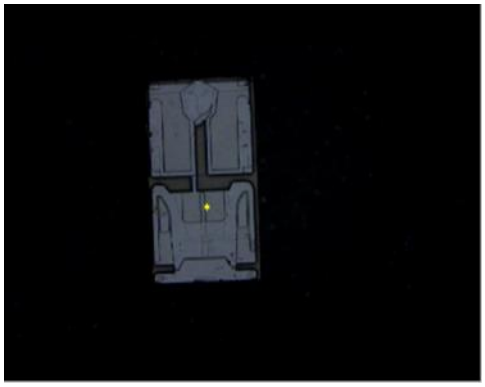
Figure 5.17(A) 1st time particle can detect slider correctly 55th frame (B), (D), (F), (H) Mean of particles 55th, 90th, 150th, 288th frame (C), (E), (G) particle can detect slider correctly 90th, 150th, 288th frame (standard deviation 50 noise level values 100)

Table 5.7 The results of the frame number that first detect and the last frame that can detect the object correctly when the standard deviation is 100.

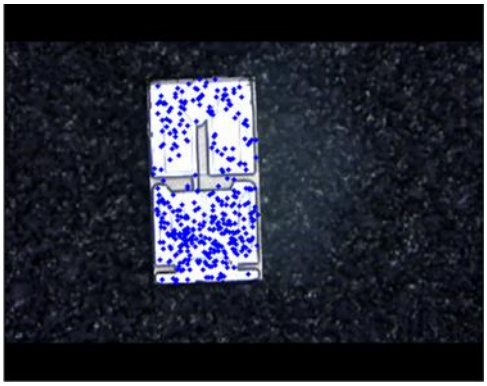
Noise movement level values	0		25	
Frame order	Frame Detect	Last Frame detect	Frame Detect	Last Frame detect
Test No.1	62nd	283rd	69th	278th
Test No.2	62nd	283rd	69th	278th
Test No.3	62nd	283rd	69th	278th
Test No.4	62nd	282nd	69th	278th
Average	62nd	283rd	69th	278th
Noise movement level values	50		100	
Frame order	Frame Detect	Last Frame detect	Frame Detect	Last Frame detect
Test No.1	68th	269th	68th	126th
Test No.2	68th	268th	68th	125th
Test No.3	69th	269th	68th	126th
Test No.4	68th	269th	69th	126th
Average	68th	269th	68th	126th



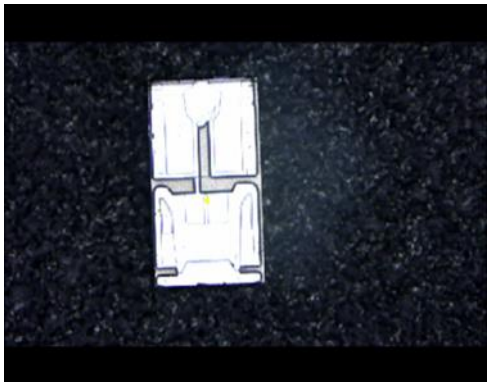
(A)



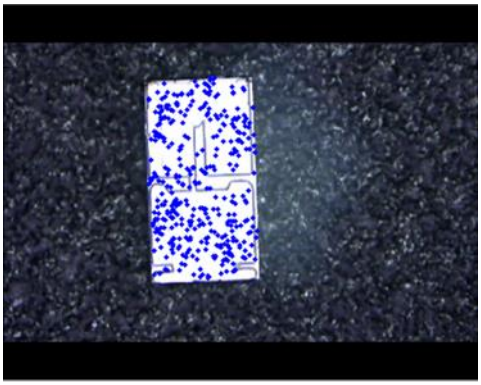
(B)



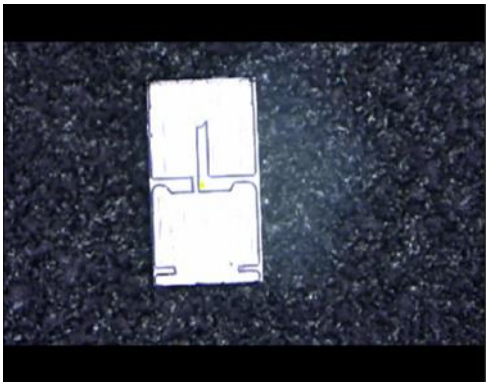
(C)



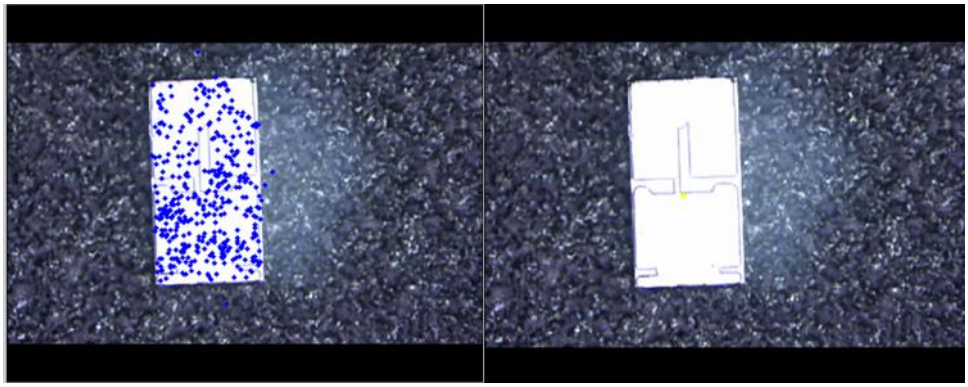
(D)



(E)

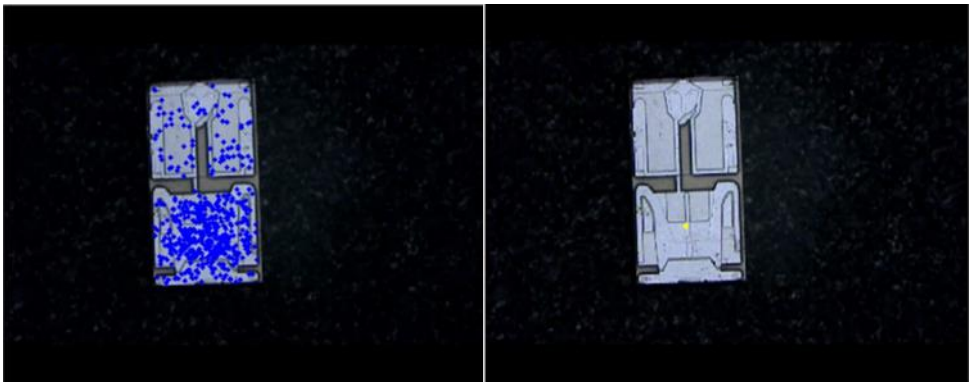


(F)

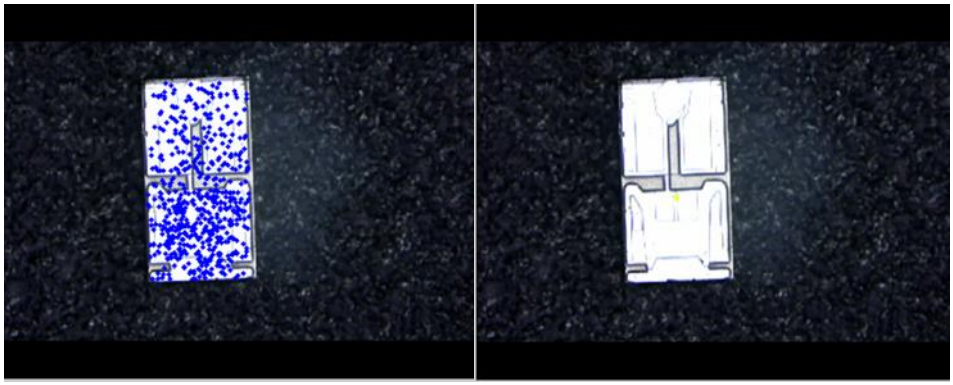


(G) (H)

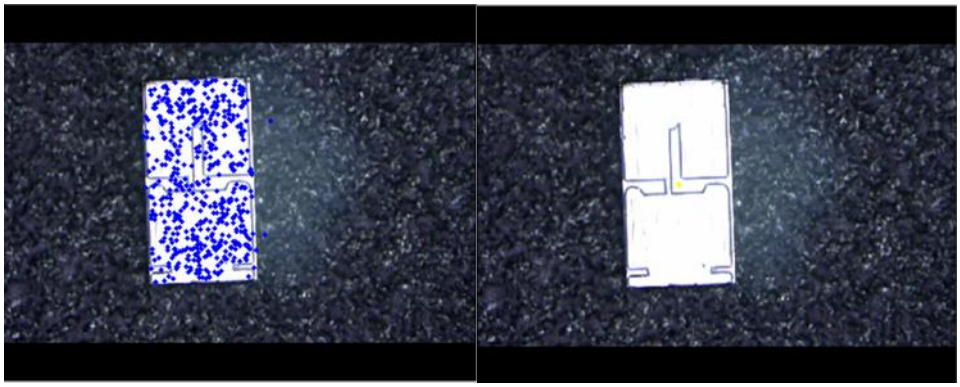
Figure 5.18 (A) 1st time particle can detect slider correctly 62nd frame
(B), (D), (F), (H) Mean of particles 62nd, 90th, 150th, 283rd frame
(C), (E), (G) particle can detect slider correctly 90th, 150th, 283rd frame
(standard deviation 100 noise level values 0)



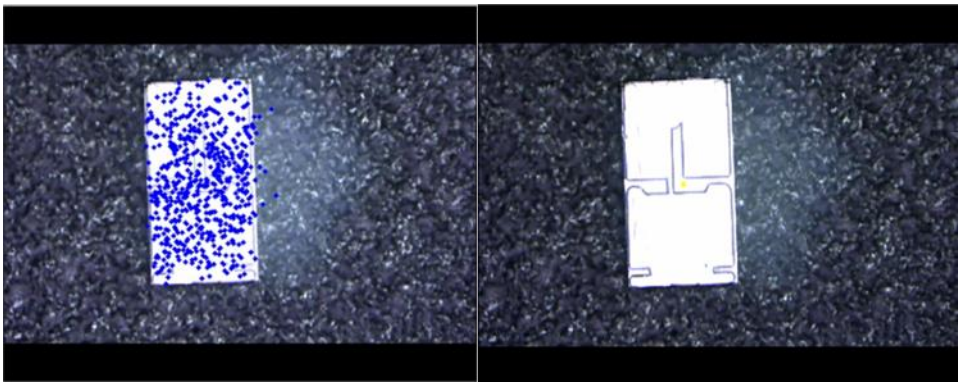
(A) (B)



(C) (D)

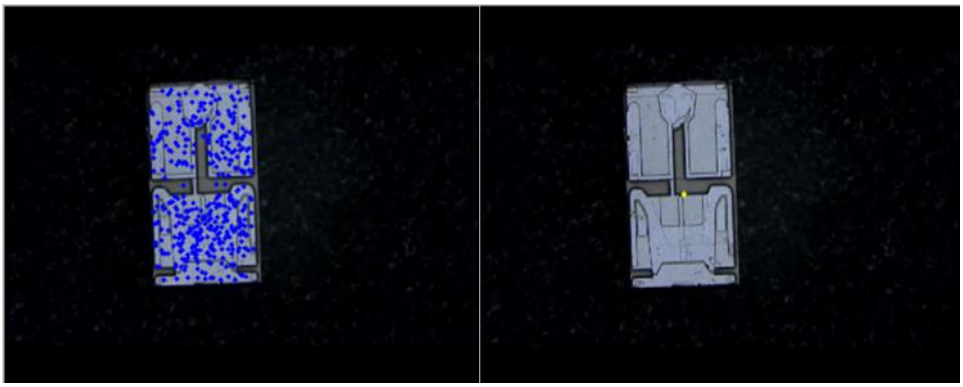


(E) (F)

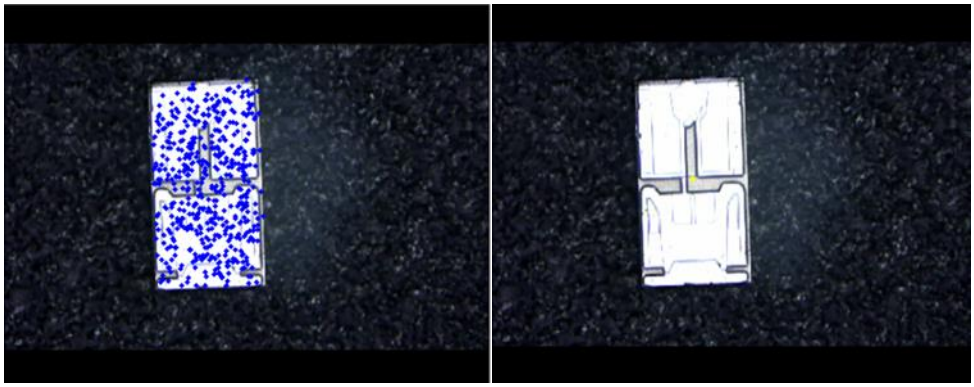


(G) (H)

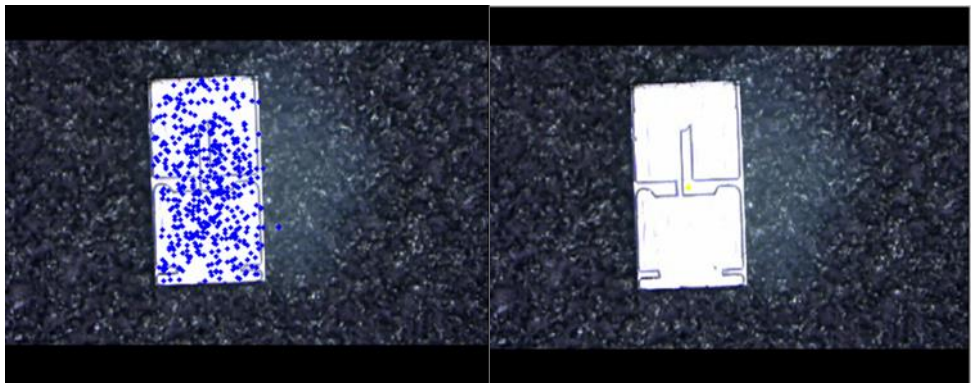
Figure 5.19(A) 1st time particle can detect slider correctly 69th frame
(B), (D), (F), (H) Mean of particles 69th, 90th, 150th, 278th frame
(C), (E), (G) particle can detect slider correctly 90th, 150th, 278th frame
(standard deviation 100 noise level values 25)



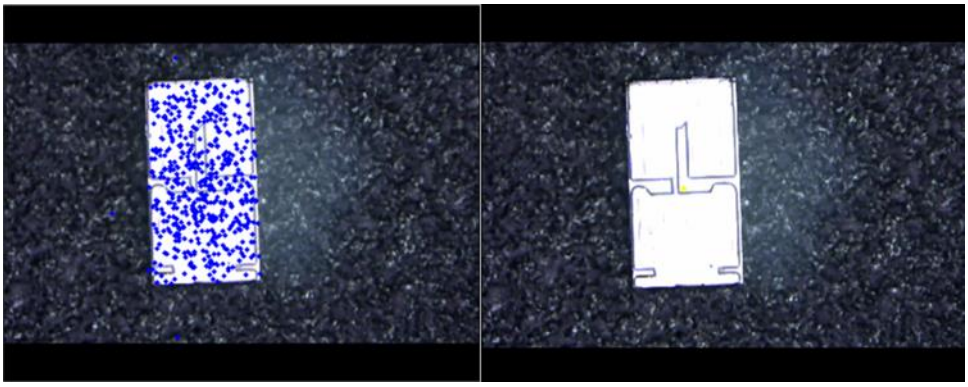
(A) (B)



(C) (D)

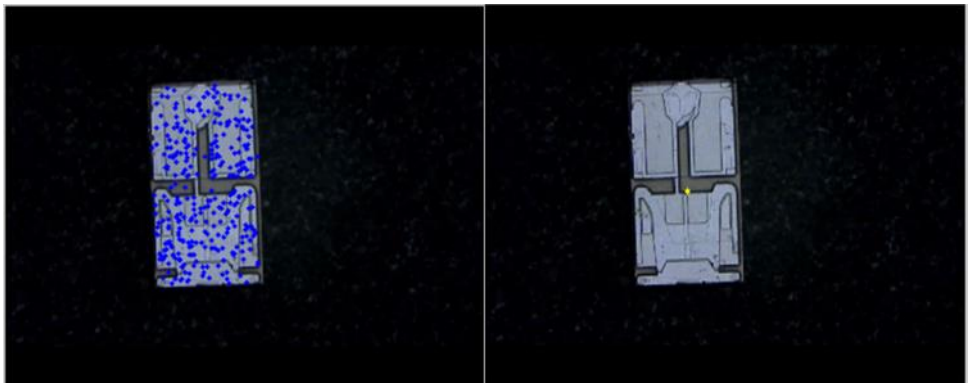


(E) (F)

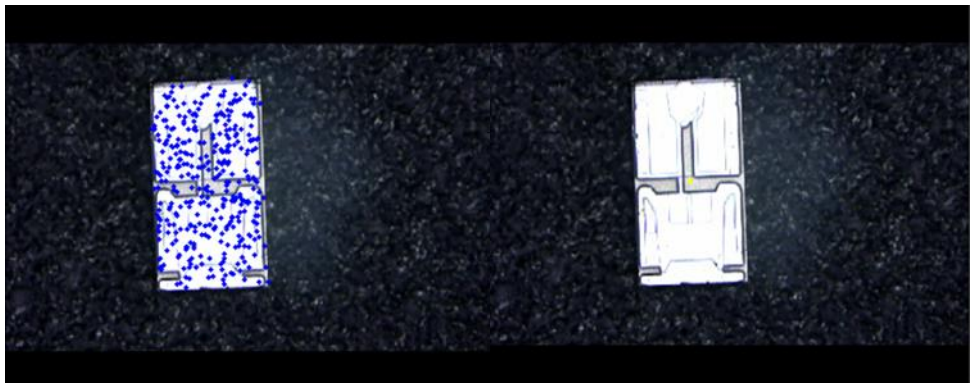


(G) (H)

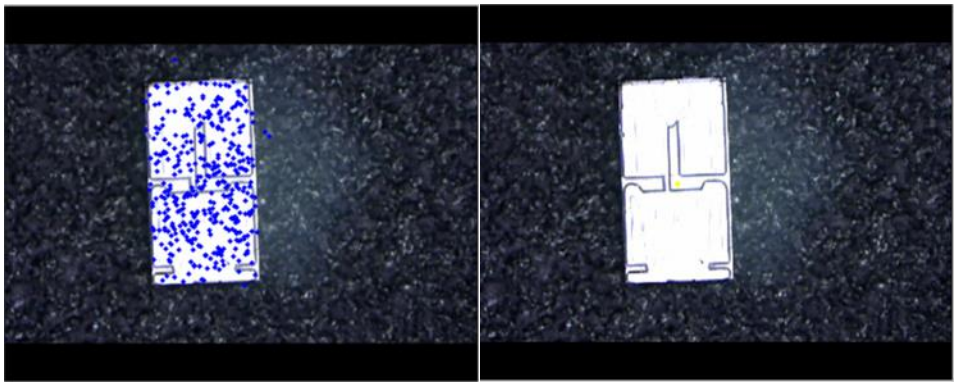
Figure 5.20 (A) 1st time particle can detect slider correctly 68th frame
(B), (D), (F), (H) Mean of particles 68th, 90th, 150th, 269th frame
(C), (E), (G) particle can detect slider correctly 90th, 150th, 269th frame
(standard deviation 100 noise level values 50)



(A) (B)



(C) (D)

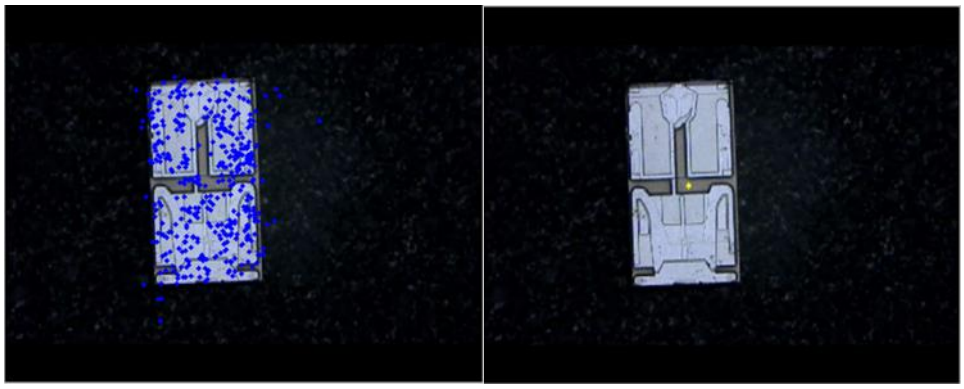


(E) (F)

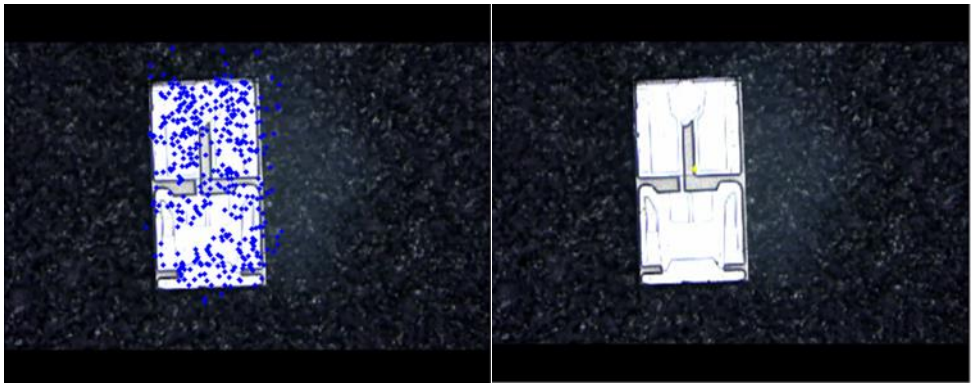
Figure 5.21 (A) 1st time particle can detect slider correctly 68th frame
(B), (D), (F) Mean of particles 68th, 90th, 126th frame
(C), (E) particle can detect slider correctly 68th, 126th frame
(standard deviation 100 noise level values 100)

Table 5.8 The results of the frame number that first detect and last detect the object correctly when the standard deviation is 200.

Noise movement level values	0		25	
Frame order	Frame Detect	Last Frame detect	Frame Detect	Last Frame detect
Test No.1	-	-	-	-
Test No.2	--	-	--	-
Test No.3	-	-	-	-
Test No.4	-	-	-	-
Average	-	-	-	-
Noise movement level values	50		100	
Frame order	Frame Detect	Last Frame detect	Frame Detect	Last Frame detect
Test No.1	-	-	-	-
Test No.2	--	-	--	-
Test No.3	-	-	-	-
Test No.4	-	-	-	-
Average	-	-	-	-

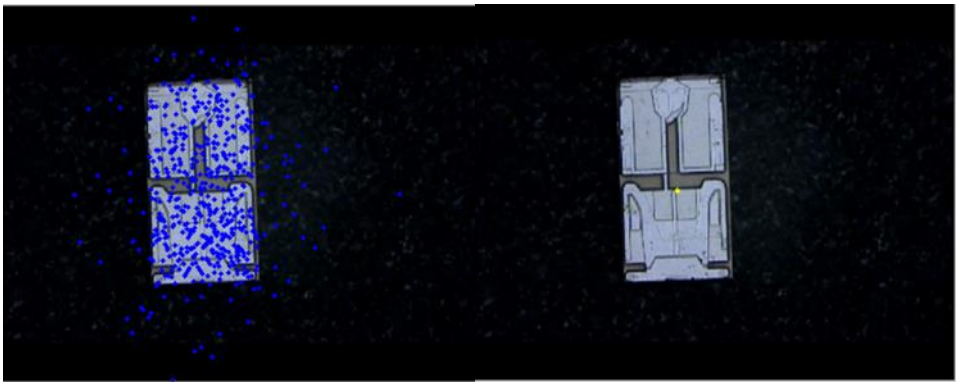


(A) (B)

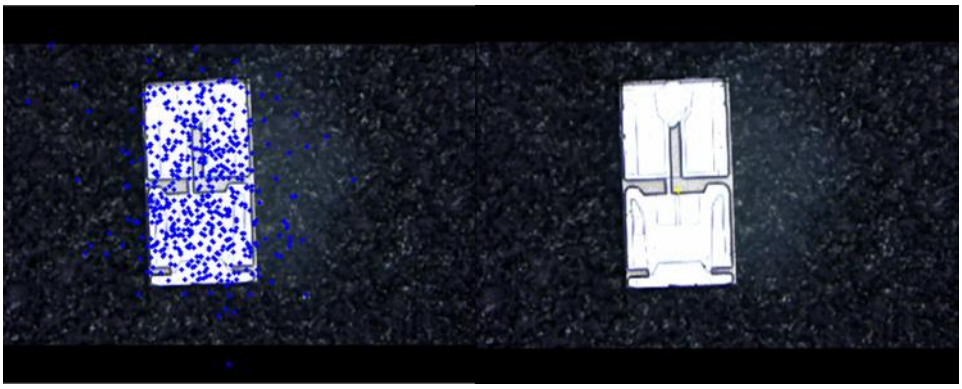


(C) (D)

Figure 5.22 (A), (C), Particle cannot detect slider correctly 68th, 90th frame
(B), (D) Mean of particles 68th, 90th frame
(standard deviation 200 noise level values 0)

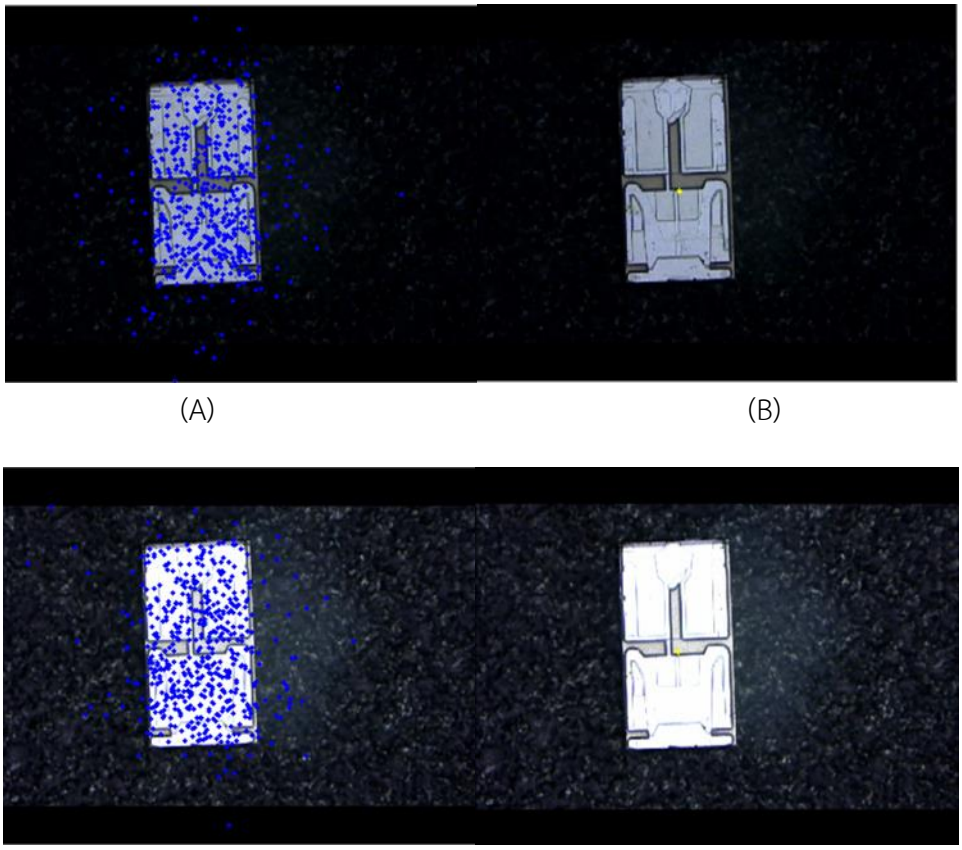


(A) (B)



(C) (D)

Figure 5.23 (A), (C), Particle cannot detect slider correctly 68th, 90th frame
(B), (D) Mean of particles 68th, 90th frame
(standard deviation 200 noise level values 25)



(A) (B)

(C) (D)

Figure 5.24 (A), (C), Particle cannot detect slider correctly 68th, 90th frame (B), (D) Mean of particles 68th, 90th frame (standard deviation 200 noise level values 50)

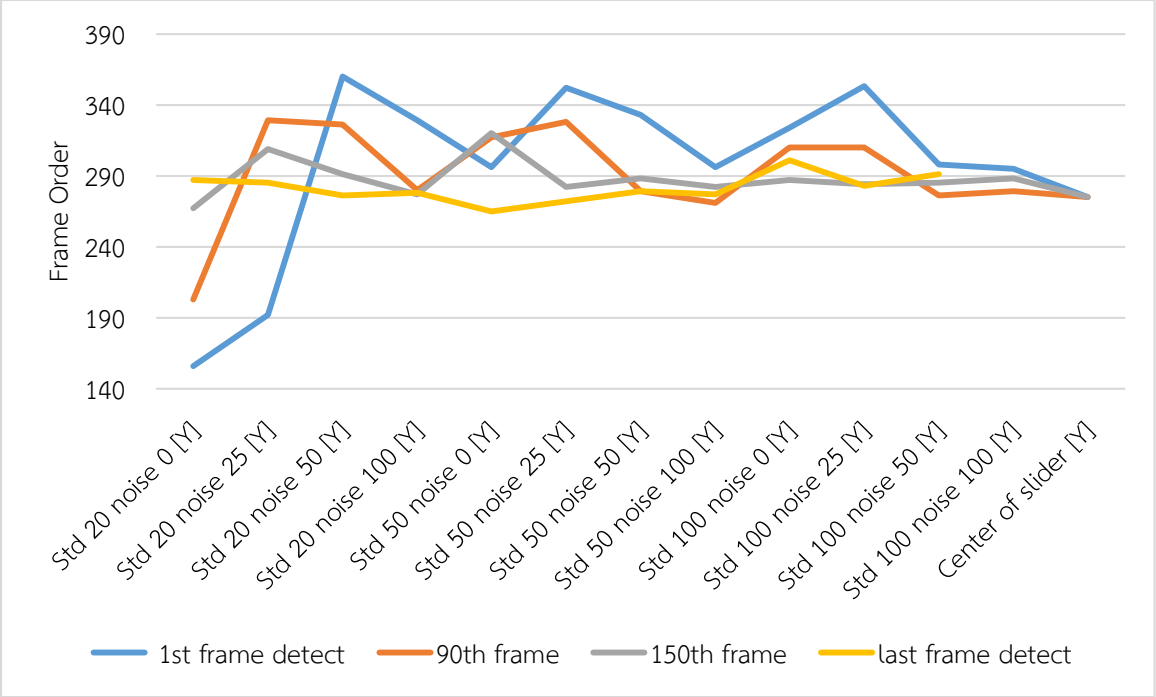


Figure 5.25 Line chart compare mean of particle position (Y)

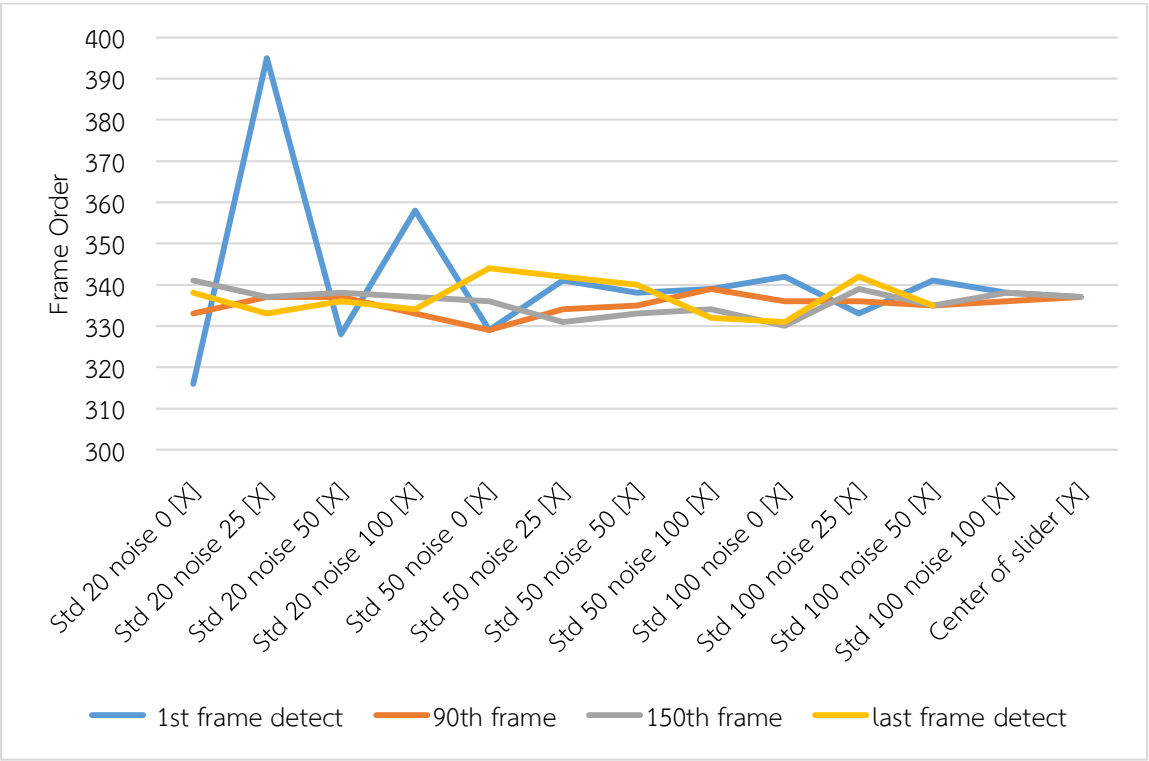


Figure 5.26 Line chart compare mean of particle position (Y)

Mean of the particle represent the slider position, in Y and X direction the position have many variation when standard deviation 20.

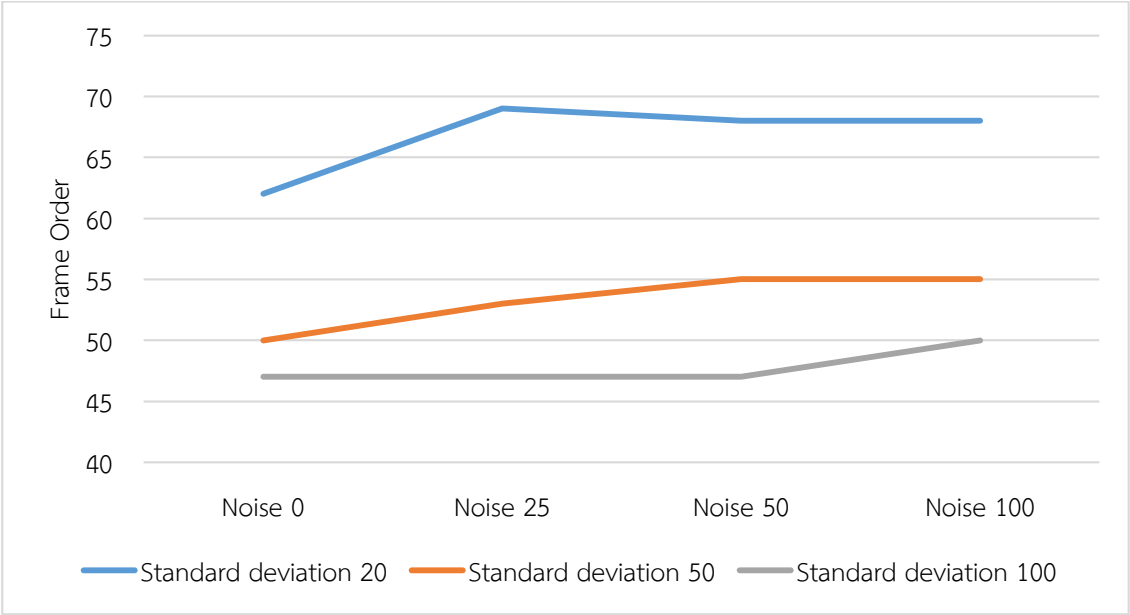


Figure 5.27 Line chart compare 1st time that particle can detect slider correctly

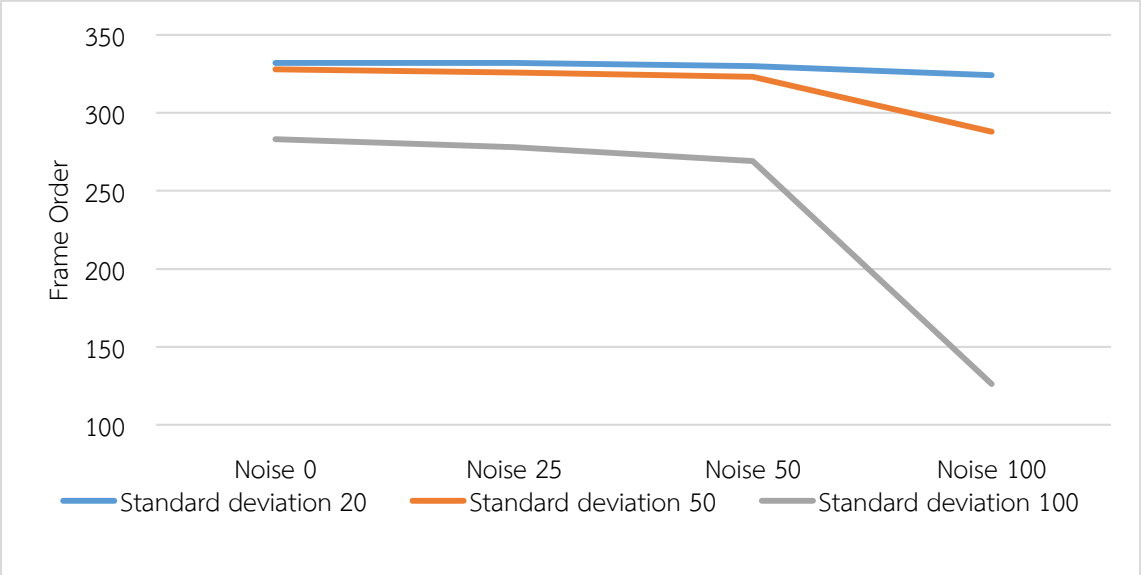


Figure 5.28 Line chart compare last time that particle can detect slider correctly

This experiment show that the standard deviation and noise of the movement level values are inverse variation with robustness of environment and the standard deviation 200 still cannot detect slider accurately.

The last experiment is run on the same condition of slider machine (same lighting ,moving and positioning. The followings are the parameters attempted to be adjusted in the experiments.

- Standard deviation σ = 20, 50, 100
- Noise level values (E) = 0 (basic particle filter), 25, 50, 100

Table 5.9 Standard deviation 20 machine condition

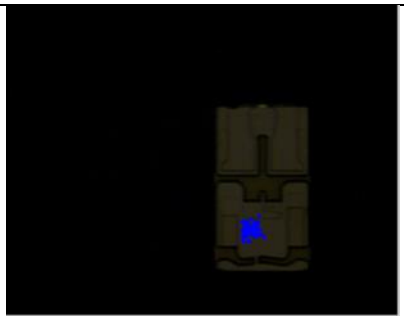
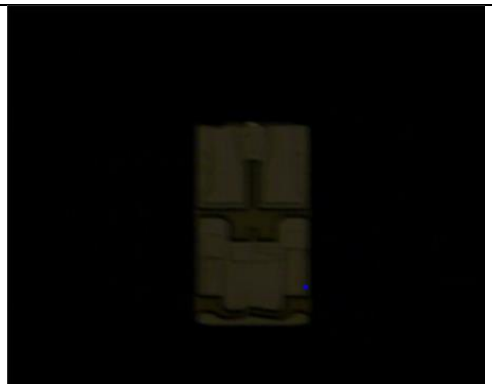
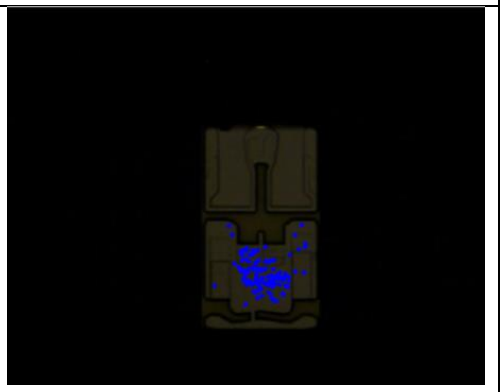
Noise level values	0	0
Detect slider (frame order)	 214th	 216th
Noise level values	25	25
Detect slider (frame order)	 209th	 210th

Table 5.9 Standard deviation 20 machine condition (CONT.)

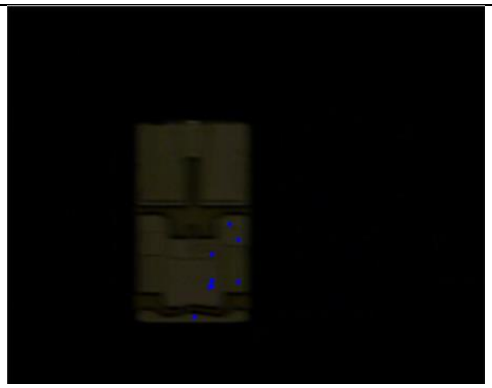
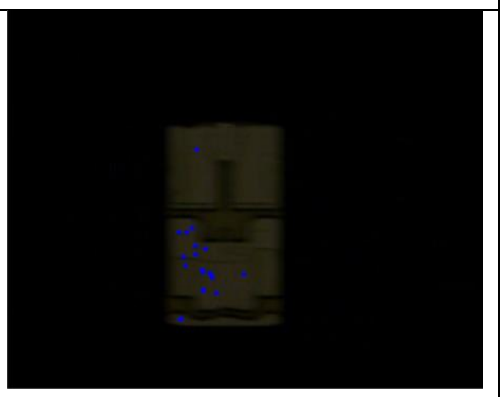
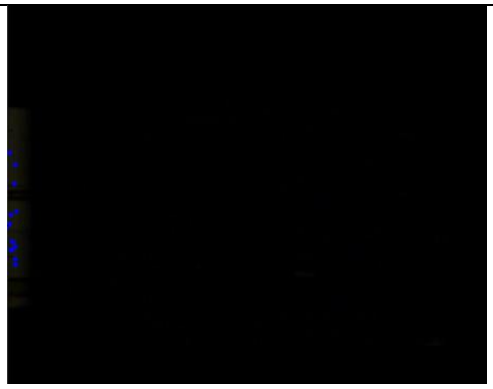
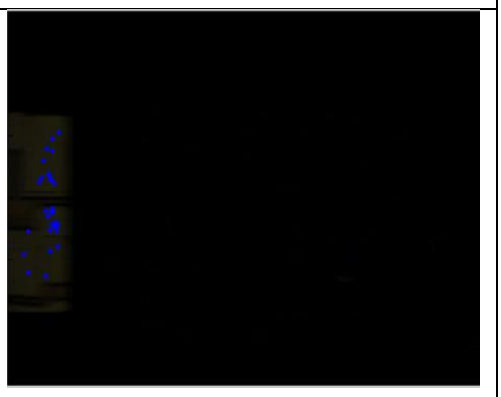
Noise level values	50	50
Detect slider (frame order)	 205th	 207th
Noise level values	100	100
Detect slider (frame order)	 177th	 179th

Table 5.10 Standard deviation 50 machine condition

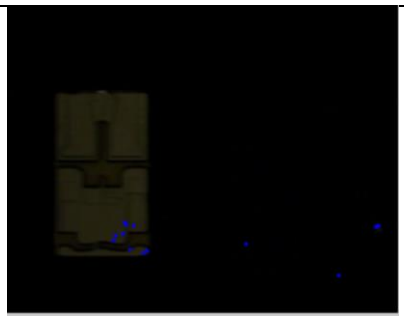
Noise level values	0	0
Detect slider (frame order)	 184th	 185th

Table 5.10 Standard deviation 50 machine condition (CONT.)

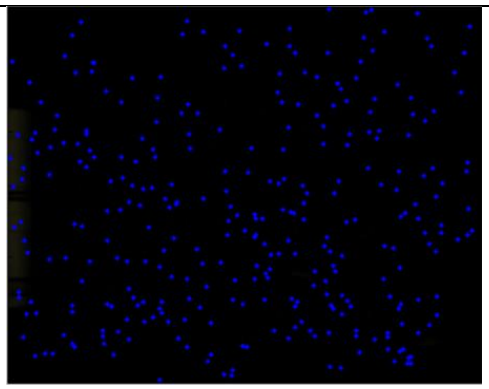
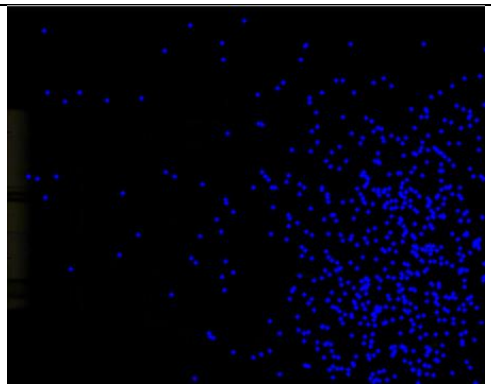
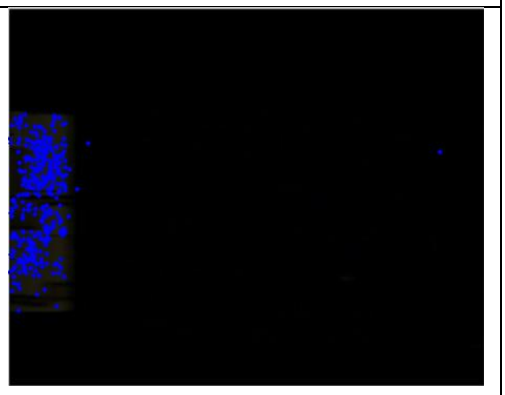
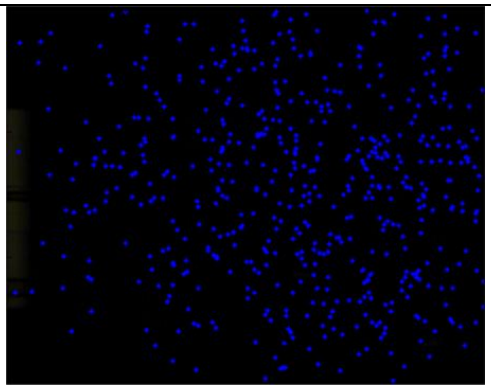
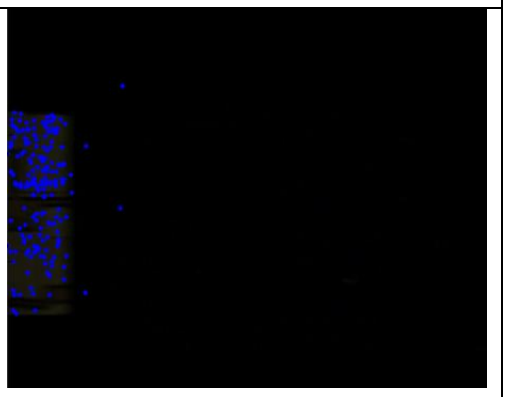
Noise level values	25	25
Detect slider (frame order)	 177th	 179th
Noise level values	50	50
Detect slider (frame order)	 177th	 179th
Noise level values	100	100
Detect slider (frame order)	 177th	 179th

Table 5.11 Standard deviation 100 machine condition

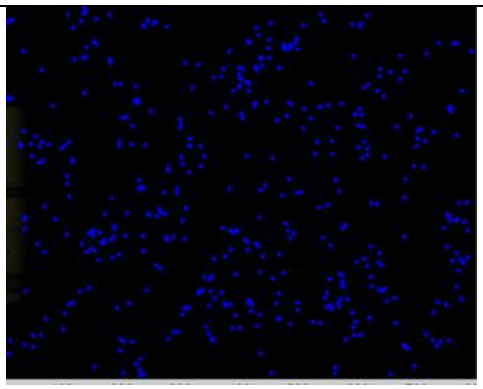
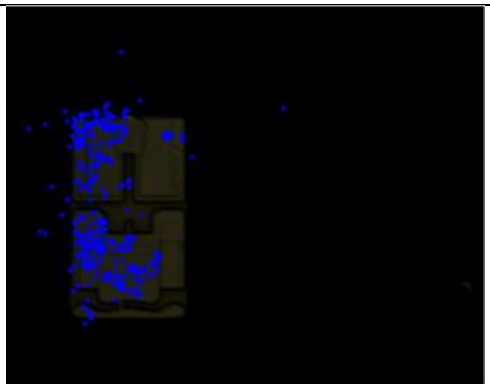
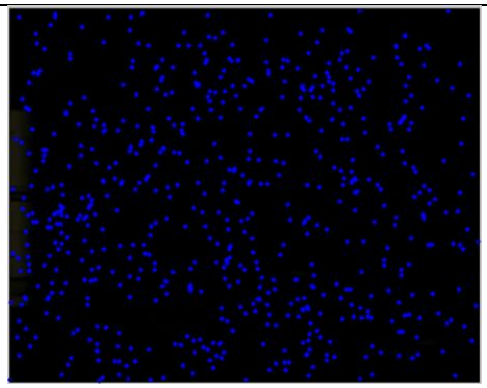
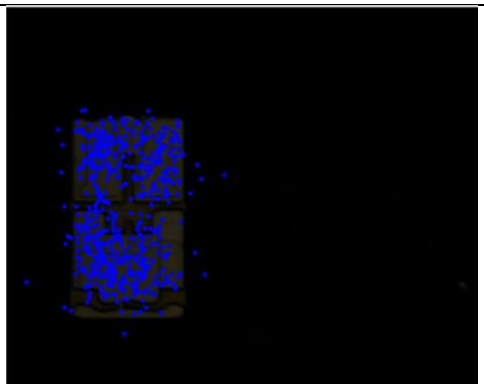
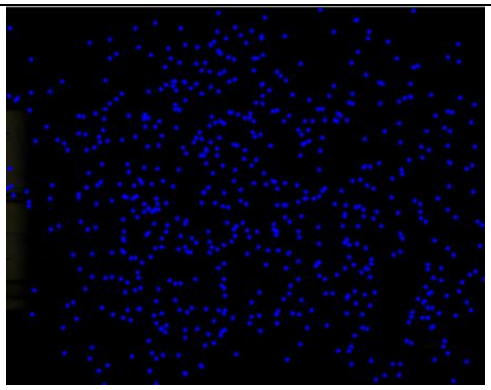
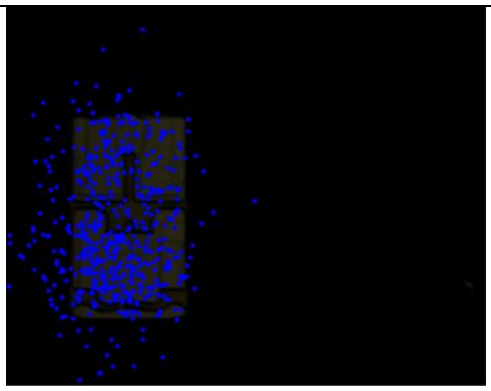
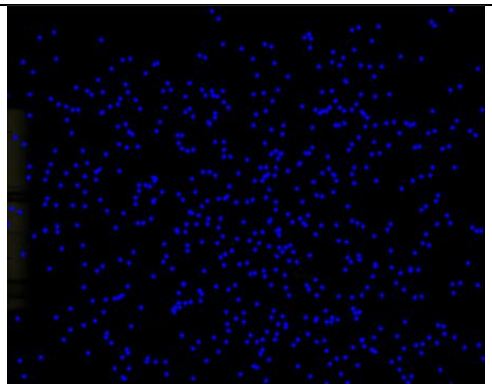
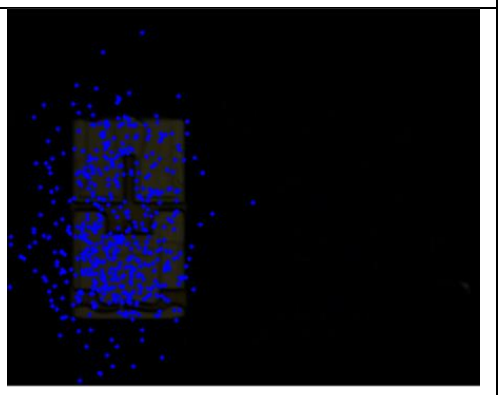
Noise level values	0	0
Detect slider (frame order)	 177th	 186th
Noise level values	25	25
Detect slider (frame order)	 177th	 186th
Noise level values	50	50
Detect slider (frame order)	 177th	 186th

Table 5.11 Standard deviation 100 machine condition (CONT.)

Noise level values	100	100
Detect slider (frame order)	 177th	 186th

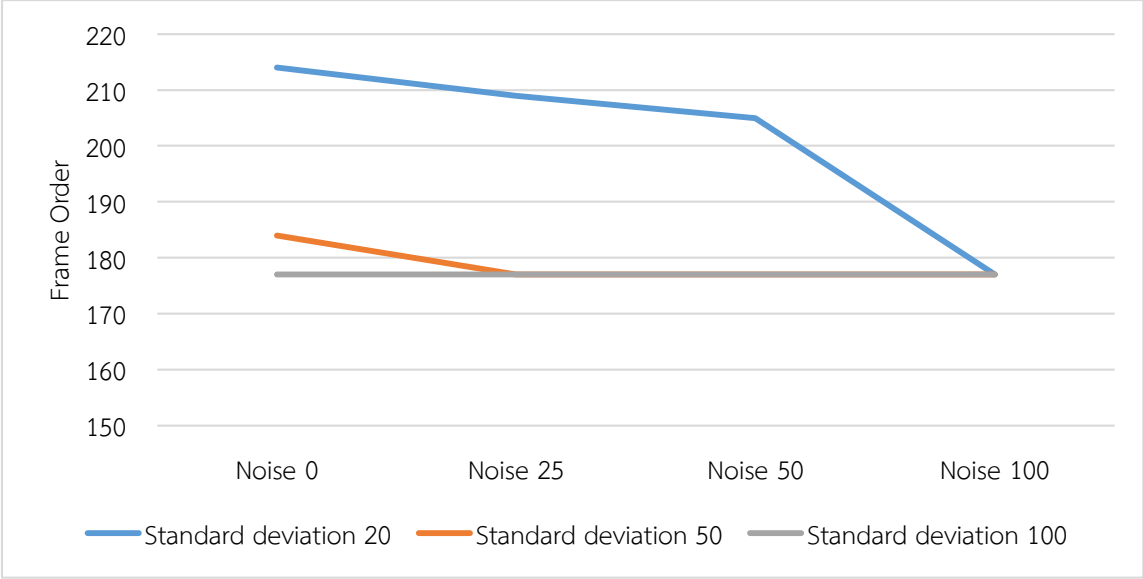


Figure 5.29 Line chart compare 1st time that particle can detect moving slider in machine condition

In the machine condition, standard deviation 50 is the best parameter that can detect slider fastest and noise level values 25 is the most accuracy

CHAPTER 6
CONCLUSION

6.1 Conclusion

In this paper, the proposed vision system with the particle filter to detect the interesting slider, ABS, is applied. The proposed algorithm is applied on the field prototype developed at the Control and Automation Research Unit Laboratory. The performance of the proposed system is verified by applying to the real process In summary, a large number of particles can detect the object faster; however, computational time will be increased (more sampling more calculating)when the standard deviation increases. Too small standard deviation generates a small number of particles with satisfied likelihood so it needs more re-sampling.

From the 1st experiments, there is flare in the center of video the result show that when slider not on screen the particle detect flare. More standard deviation and high noise level values can detect slider faster but the standard deviation 200, the particles cannot detect moving slider correct (there are some particles detect background)

From the 2nd experiments, to test the robustness to the light (slider always on the screen then turn on and increase the light. The standard deviation 200 cannot accuracy detect slider from the start, and standard deviation 100 and 25 are less robustness than standard deviation 50, standard deviation 100 can detect slider as fast as stand deviation 50 but it less accuracy and standard deviation 25 is detect slider slower than standard deviation 50. The robustness of noise movement level values 0, 25 and 50 are almost the same but noise level values 50 can detect slider faster when have flare on the lens.

The basic particle filter (noise movement level = 0) cannot detect slider faster but there is less robustness than the method that we increase noise movement level to 25 and 50

The mean of the slider position has many variation when the standard deviation is 20

The last experiments is shown that in machine condition basic particle filter cannot detect moving slider fast enough and particles are spread in small area and standard deviation 100, the particles are spread out of the slider

So the parameter that should use on feasibility of the machine condition are particle number 1000, standard deviation 50 and noise level values 50.

REFERENCES

[1] Fazlina Ahmat Ruslan, Zainazlan Md Zain, Ramli Adnan and Abd Manan Samad, “Flood Water Level Prediction and Tracking Using Particle Filter Algorithm,” IEEE 8th International Colloquium on Signal Processing and its Applications, 2012.

[2] Fredrik Gustafsson, Fredrik Gunnarsson, Niclas Bergman, Urban Forssell, Jonas Jansson, Rickard Karlsson,and Per-Johan Nordlund, “Particle Filters for Positioning, Navigation and Tracking” I*EEE Transactions on Signal Processing,2002*

[3] George Poyiadjis, Arnaud Doucet and Sumeetpal S. Singh, “Maximum Likelihood Parameter Estimation in General State-Space Models using Particle Methods”, 2005

[4] TIAN-Zengshan and LUO Lei, “Particle Filter positioning and tracking Based on dynamic model,” IEEE Conference on Automation Science and Engineering, 2008.

[5] Jiayan Li, Xiaofeng Lu, Lianyi Ding and Hengli Lu, ”Moving Target Tracking via Particle Filter Based on Color and Contour Features”, Information Engineering and Computer Science (ICIECS), 2010 2nd International Conference

[6] Peiliang Wu, Lingfu Kong, Fengda Zhao, Xianshan Li, “Particle Filter Tracking Based on Color and SIFT Features” Audio, Language and Image Processing, 2008. ICALIP 2008. International Conference, pp. 932 - 937

[7] Jiang LI, Chin-Seng CHUA,”TRANSDUCTIVE INFERENCE FOR COLOR-BASED PARTICLE FILTER TRACKING”, Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference, pp.949-952

[8] M. Isard and A. Blake, “Conditional density propagation for visual tracking,” International Journal of Computer Vision, vol. 29 (1), pp. 5-28, 1998.

[9] P. Jensfelt et al, “Feature based condensation for mobile robot localization,” Proceeding of the IEEE International Conference

[10]Jiang LI, Chin-Seng CHUA, “Transductive inference for color-base particle filter tracking,” Proceeding of the IEEE International Conference on Image Processing, 2003.K.Elissa, “Titleofpaperifknown,” unpublished.

[11]XU Kehu and LI Ke, “Optimize Particle Filter Tracking algorithm By Features Fusion and Variable Noise Covariance,” IEEE International Conference on Mechatronic Science, Electric Engineering and Computer, 2011.

[12]Jin Li, Hong Yu, Lulu Zhou, Hong Liang, and Lei Wang, “An Adaptive Unscented Particle Filter Tracking Algorithm Based on Color Distribution and Wavelet Moment,” IEEE International Conference on Industrial Electronics and Applications, 2007.

[13]Azimi—Sadjadi B. and Krishnaprasad P. S. Change, “Detection for Nonlinear Systems : A Particle Filtering Approach (A),” Proceedings of the 2002 American Control Conference, 2002.

APPENDIX A

PUBLICATION PAPER

ON THE FLY VISION SYSTEM FOR AIR BEARING SURFACE INSPECTION

Thanee Samsicharoenlap
Data Storage Technology,
King Mongkut's Institute of Technology Ladkrabang
Bangkok, Thailand

Somyot Kaitwanidvilai
Faculty of Engineering,
King Mongkut's Institute of Technology Ladkrabang
Bangkok, Thailand

***Abstract*— Now a day, the improvement in manufacturing process in the hard disk drive industry is required to achieve the advanced manufacturing process for nano technology. Vision system has been extensively used in machine inspection and monitoring. Especially, a very small object such as HDD components, air bearing surface (ABS), slider, etc. needs a more reliable system which is difficult to be achieved by human inspection. In addition, the fast speed of inspection is also required. In this paper, the development of an automatic visual system for detecting the position of the air bearing surface (ABS) on a slider wafer is described. The non-stop vision system called "vision on the fly" is proposed to enhance the productivity in terms of unit per hour (UPH). The adaptive particle filter technique is adopted to determine the position of the interesting object even the object is moving. Based on the results indicated, more than 20% of UPH can be achieved by the proposed system.**

***Index Terms*—vision on the fly, Air Bearing Surface, automatic visual inspect.**

I. INTRODUCTION

Presently, vision system has been widely utilized in the manufacturing process for many kinds of applications such as positioning, inspection, etc. As shown in the previous research works [1-4], vision system has been continuously improved by many proposed algorithms to achieve higher accuracy and speed of the inspection. However, for extremely precise applications, the interesting objects should be stopped at the inspection point to achieve high accuracy inspection system. The production rate may be reduced from the stopping time needed for inspection machine. For example, the inspection of slider wafer needs about 1.5 seconds for stopping the pallet before performing the visual inspection. The machine vision system needs to wait until the slider wafer completely stops before capturing the image, spending more time of inspection. To increase the inspection speed, this research proposes a new visual inspection system for detecting ABS by using particle filter. The proposed system can reduce the inspection time by using on the fly vision; in the other words, the unit per hour (UPH) of the machine is increased.

Due to the very small size of the slider, the moving distance is very short (2-3 mm) which we can assume that the moving is in linear envelopment. Particle filter can be applied by using this assumption. Particle filter has been applied to many applications such as visual tracking [1], mobile robot localization [2], color-base tracking [3], tracking algorithm by features fusion and the Variable Noise Covariance [4]. All techniques are adopted to track the moving objects under the variation of environmental conditions such as light intensity, high speed movement, etc.

II. ABS AND PARTICLE FILTER THEORY

Air passing under the air bearing surfaces (ABS) provides the required "lift" across the entire disk surface. ABS design has to take into account the weight, velocity, and skew to achieve a uniformed flying height. Fig. 1 shows the typical ABS used in hard disk. In the slider attachment process, the position of ABS needs to be inspected before moving the slider and putting on the suspension for attachment.

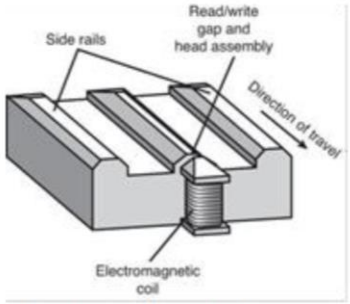
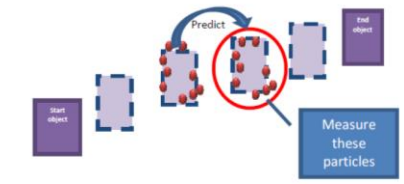
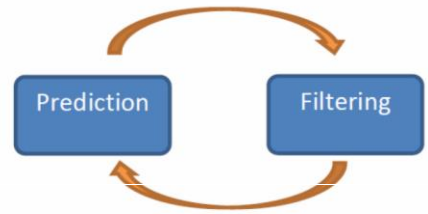


Fig. 1 Typical ABS on the slider

The particle filter technique adopted in this research work is based on the sequential Monte Carlo method (SMC) and starts by partitioning the state space in various parts. Then, based on probability measure, the particle positions can be assigned. The concentration of the particles depends on the probability measure. As the probability measure increases, the concentration of the particles also increases [5]. Based on this theory, the prediction of the object's trajectory can be determined by solving the differential equation. Particle filter is a state estimation method using a lot of particles. The two main steps for achieving the particle are prediction and filtering. Prediction is carried out by solving the differential equation; the filtering is done by re-selection of the particles according to their likelihood.



(a)



(b)

Fig. 2 Basis of the technique, particle filter (a) application on image processing, (b) basic steps of the particle filter technique

Following details describe the basis of a particle filter on image processing [4-5].

The state of a tracked object is described by the vector X_n while the vector in denotes as all the observations $\{Y_1, \ldots, Y_T\}$ up to time T. Monte Carlo approach is used for generating many articles (N). Each of N samples has the state $X_n^{(i)}$ and its weight $W_n^{(i)}$ The core idea is to use the weighted addition of a random sample to describe the required posterior probability density and get the state estimate value.
The general state space model is

System model: $X_n = f(X_{n-1}) + W_n$ (1)
Observation model : $Y_n = g(Y_{n-1}) + V_n$ (2)

W_n denotes the system noise ;
V_n denotes the observation noise ;

Suppose $X_{0:n}$ {X = X(0),X(1),...,X (n)}
$Y_{1:n}$ {Y = Y(1),Y(2),...,Y(n)}, the probability distribution of a certain time in 0 to n, $P(X_{0:n})$ can be expressed as:

$$P(X_{0:n}) = \{X_{0:n}^{(i)}, w_n^{(i)}\}_{i-1}^{N} \quad (3)$$

$X_{0:n}^{(i)}$ denotes the N particle of a certain time in 0 to n;
$W_n^{(i)}$ denotes Normal weight of N particle;
$\sum_{n=1}^{N} w_n^{(i)} = 1$: N is the number of particles.

The posterior probability density in n moment can be denoted by discrete weights:

$$P(X_{0:n}|Y_{1:n}) \approx \sum_{n=1}^{N} w_n^{(i)} \delta\{X_{0:n} - X_{0:n}^{(i)}\} \quad (4)$$

Generate the particle samples which follow the filter distribution:

$$P(X_n|Y_{1:n}) = \int \ldots \int P(X_{0:n}|Y_{1:n})\, dX_{n-1} \ldots dX_0 \quad (5)$$

Likelihood

For tracking the object, the likelihood of a pixel's color of each particle and target color has many advantages. Comparing their color and finding their likelihood can be used to achieve the likelihood function:

$$P\left(Y_n \middle| X_{n|n-1}^{(i)}\right) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{d^2}{2\sigma^2}} \quad (6)$$

Denote: $d = \sqrt{(R-r)^2 + (G-g)^2 + (B-b)^2}$ (7)

R, G, B denote the pixel's colors.
r, g, b denote the target colors.

Resampling

The likelihood ratio in (8) can be used for resampling the particle in the next iteration:

$$\frac{P\left(Y_n \middle| X_{n|n-1}^{(i)}\right)}{\sum_k P\left(Y_n \middle| X_{n|n-1}^{(i)}\right)} \quad (k = 1, \ldots, N) \quad (8)$$

Next, the new position of particle can be determined by (9).

$$\begin{pmatrix} X_n \\ Y_n \\ X'_n \\ Y'_n \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X_{n-1} \\ Y_{n-1} \\ X'_{n-1} \\ Y'_{n-1} \end{pmatrix} + \begin{pmatrix} V_x \\ V_y \\ V_{x'} \\ V_{y'} \end{pmatrix} \quad (9)$$

III. Experimental Results

In this research work, particle filter is applied to inspect the ABS on the slider at the slider attachment machine. To verify the effectiveness of the proposed algorithm, the performance in terms of the number of the object detection frame was investigated. In addition, noise is added to the system for verifying the robustness of the proposed algorithms. In the experiments, the test by adjusting the parameters σ in (6), the particle number and noise metric in (9) were performed. In the video stream from the vision system, the slider has completely been appeared on screen at the 44th of the video frame. The followings are the parameters attempted to be adjusted in the experiments.

- Standard deviation σ = 20, 50, 100, 200
- Number of particles N = 100, 1000
- Noise level values = 25, 50, 100
- Number of repeated experiments = 4

Table 1 The results of the frame number that first detect the object correctly when the number of particles is 100 and noise is 25.

σ	Test No.	20	50	100	200
The first frame that particle can find the object	1	69	68	66	54
	2	67	67	69	50
	3	70	67	68	52
	4	69	69	69	58
	Average	68	67	68	53

Table 2 The results of the frame number that first detect the object correctly when the number of particles is 100 and noise is 100.

σ	Test No.	20	50	100	200
The first frame that particle can find the object	1	50	51	46	44
	2	49	48	47	44
	3	50	46	47	44
	4	50	50	46	45
	Average	49	48	46	44

The results in Tables 1 and 2 indicate that the large number of particles can detect the object faster. When one particle found the object, the likelihood of the color is much increased. This makes all particles attempts to move to the object position in the next frame. In addition, increasing the variance will help to detect the object faster in this application. Fig. 3 shows the detection when there is no slider on the screen. As seen in this figure, particles detect the noise of light instead when there is no slider on the screen.

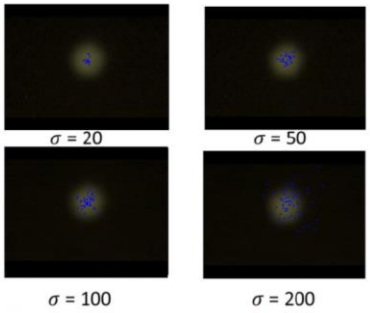


Fig. 3 No slider on the inspection area: N = 100, noise = 25

Figs. 4 and 5 show the captured images when the particle firstly finds the object at various standard deviations. The blue dots represent the positions of particles.

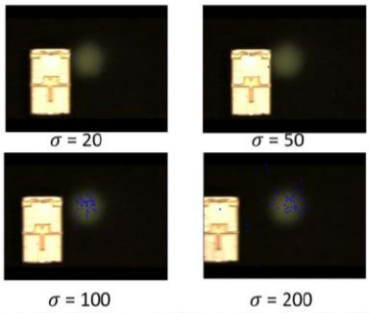


Fig. 5. The first frame that the particle finds the slider : N = 100, noise = 25

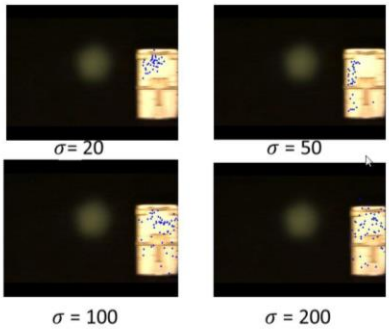


Fig. 6. Resampling after the particles find the object on the screen where N = 100, noise = 25

IV. CONCLUSION

In this paper, the proposed vision system with the particle filter to detect the interesting object, ABS, is applied. The proposed algorithm is applied on the field prototype developed at the Control and Automation Research Unit Laboratory which is financially supported by Seagate (Thailand) Technology Co. Ltd. The performance of the proposed system is verified by applying to the real process In summary, a large number of particles can detect the object faster; however, computational time will be increased when the standard deviation increases. Too small standard deviation generates a small number of particles with satisfied likelihood so it needs more re-sampling.

ACKNOWLEDGEMENTS

This research work is supported by King Mongkut's Institute of Technology Research Fund., Seagate (Thailand) Co. Ltd., NECTEC, NSTDA and DSTAR, KMITL.

REFERENCES

[1] M. Isard and A. Blake, "Conditional density propagation for visual tracking," International Journal of Computer Vision, vol. 29 (1), pp. 5-28, 1998.

[2] P. Jensfelt et al, "Feature based condensation for mobile robot localization," Proceeding of the IEEE International Conference

[3] Jiang LI, Chin-Seng CHUA, "Transductive inference for color-base particle filter tracking," Proceeding of the IEEE International Conference on Image Processing, 2003.K. Elissa, "Title of paper if known," unpublished.

[4] XU Kehu and LI Ke, "Optimize Particle Filter Tracking algorithm By Features Fusion and Variable Noise Covariance," IEEE International Conference on Mechatronic Science, Electric Engineering and Computer, 2011.

[5] Fazlina Ahmat Ruslan, Zainazlan Md Zain, Ramli Adnan and Abd Manan Samad, "Flood Water Level Prediction and Tracking Using Particle Filter Algorithm," IEEE 8th International Colloquium on Signal Processing and its Applications, 2012.

[6] Jin Li, Hong Yu, Lulu Zhou, Hong Liang, and Lei Wang, "An Adaptive Unscented Particle Filter Tracking Algorithm Based on Color Distribution and Wavelet Moment," IEEE International Conference on Industrial Electronics and Applications, 2007.

[7] TIAN-Zengshan and LUO Lei, "Particle Filter positioning and tracking Based on dynamic model," IEEE Conference on Automation Science and Engineering, 2008.

[8] Azimi—Sadjadi B. and Krishnaprasad P. S. Change, "Detection for Nonlinear Systems : A Particle Filtering Approach [A]," Proceedings of the 2002 American Control Conference, 2002.

AUTHOR BIOGRAPHY

Name-Surname: Mr. Thanee Samsicharoenlap

Date of Birth: October 10th, 1986

Present Address: 700, Phetkasem 92/2, Bangkaeneur, Bangkae, Bangkok, Thailand

Education: 2005-2009: Bachelor degree in Electronics Engineering, King Mongkut's institute of technology Ladkrabang.

Scholarships: 2011-2012 Scholarship for study in Master of Engineering in Data Storage Technology (English program) by NSTDA, KMITL and Seagate Technology (Thailand) Ltd.

Publications: Thanee S., Somyot K., "ON THE FLY VISION SYSTEM FOR AIR BEARING SURFACE INSPECTION", IEECON 2014, The International Electrical Engineering Congress 2014 at Pattaya, Thailand, March 19-21, 2014.

Experience:

2010-2014 Seagate Technology (Thailand) Ltd.
- Electrical engineer

2009-2010 Rohm Integrated System (Thailand) Ltd.

- Vision Engineer