Fitts correlation tracker simulation

Spring 2007 Semester

Supervisor: Arie Nakhmani

Students:
- Shai Levi 031579717
- Noam Helmer 036332351
# Table of Contents:

1 Bibliography .......................................................................................................................... 3

2 Abstract .................................................................................................................................. 3

3 Introduction ............................................................................................................................... 4
  3.1 Trackers development history ............................................................................................ 4
  3.2 Main properties of Fitts correlation tracker ......................................................................... 4

4 Basic theory and motivation ...................................................................................................... 5
  4.1 The use of Matched Filter in tracking .................................................................................. 5
  4.2 The use of Reference-Map: ................................................................................................. 6
  4.3 Mathematical proof of the correlator functions ..................................................................... 7

5 Fitts processor algorithm .......................................................................................................... 10
  5.1 The correlator block diagram .............................................................................................. 10
  5.2 Mathematical discretization of the blocks .......................................................................... 10
    5.2.1 Weight matrices ........................................................................................................... 10
    5.2.2 Scale factors ............................................................................................................... 11
    5.2.3 Raw errors .................................................................................................................. 11
    5.2.4 Normalized correlation error ....................................................................................... 11
    5.2.5 Drift compensation ...................................................................................................... 12
    5.2.6 Total correlation error ................................................................................................. 12

6 Tracking accuracy checking ...................................................................................................... 12

7 Different implementations and results ....................................................................................... 13
  7.1 "Whole Frame" method ...................................................................................................... 14
  7.2 "Gate Mask" method ........................................................................................................... 16
  7.3 "Gated Frame" method ........................................................................................................ 18
  7.4 Center of Mass addition ....................................................................................................... 22

8 Conclusions and summary ....................................................................................................... 24
  8.1 Conclusions ......................................................................................................................... 24
  8.2 Summary ............................................................................................................................... 24

# List of figures

Figure 1: Fitts correlator block diagram ....................................................................................... 10
Figure 2: basic method flow chart .............................................................................................. 13
Figure 3: "gated frame" flow chart ............................................................................................. 18
1 Bibliography

2 Abstract
In our project we studied and examined the Fitts correlation algorithm in various ways. We first implemented the algorithm flow in MATLAB. We started our simulations tests with a white Gaussian on a black background. The Gaussian shifted randomly on the background, while our tracker was trying to follow its center. We showed different approaches, like taking the whole frame, and taking gated frame and tested them to find the best approach.

Then we tested our flows on “real” movies, with human figures moving through natural background. We spotted the obstacles that disturbed our correlator to follow the target and overcame them. Lastly, we tried to integrate our algorithm flow with center-of-mass mechanism and wrote our conclusions.

We found our Fitts Correlator to be extremely reliable. With the correct approach and the right parameters (discussed later), we were able to achieve robust results. Also, we found that center-of-mass addition might improve the correlator in a few cases (also be discussed later).
3 Introduction
This project is based on “Precision Correlation Tracker via Optimal Weighting Functions” by Dr. John M. Fitts [1], and his patent [3].

3.1 Trackers development history
In the late 60's and early 70's, as processors and digital hardware made their technological progress, most correlators were based on centroids and edge tracking algorithms. The performance of such algorithms was limited by background clutter, high intensity sources (for example, bright background spot next to the bright target) and by non-trivial adjusting of the gate size and threshold levels. Most of these algorithms were unable to track objects with the non-stationary camera.

In the late 70's, John M. Fitts issued a patent [3] on Target Adaptive Correlation Digital Tracker. This tracker has proved to be the most advanced tracker yet developed. It has performed exceptionally well in a high cluttered environment – where the centroid and edge tracking algorithms showed poor results. There was no need for threshold level and the gate could be significantly larger than the object itself.

The correlation algorithms discussed receive an array of 2-D picture frames from a video stream that can be achieved by any kind of tracking sensor (e.g. surveillance camera, radar). We will show the specific algorithm for producing the correlation errors in each direction (in elevation and deflection).

3.2 Main properties of Fitts correlation tracker
- The tracking algorithm is based on correlation and matched filter principles, therefore has the property being optimal for all trackers in the sense of tracker's noise minimization, as will be explained in the next chapter.

- The tracking errors have smaller resolution than the spatial sampling, i.e., gives errors in pixel fractions.

- The tracker processor can be implemented in simple pipe-line mechanization. This improve the real time performance by allowing the processing of the current frame without loading it entirely before. Furthermore, it reduces the storage requirement

- The processor technique is not limited to optical sensors but can be applied to any tracking sensor (e.g. radar).
4 Basic theory and motivation

4.1 The use of Matched Filter in tracking

Fitts correlator uses Matched filter to minimize the jitter due to a noisy imagery. The essence of the matched filter theory is that the output signal of a Matched Filter which has as its input a noisy signal (assuming Additive White Gaussian Noise (AWGN) model) has highest SNR at some point - larger than can be provided by any other filter. SNR (Signal to Noise Ratio) is defined as the signal power (power might be an integral on the pixels values in our case) divided by the added noise power.

We begin with 1-D continuous signal case:

- Consider 1-D signal in the form: \( y(\varepsilon) = s(\varepsilon - \delta) + n(\varepsilon) \)

  Where: \( s(\varepsilon) = \) a signal of predetermined form ; \( n(\varepsilon) = \) AWGN

  i.e., the measured output is a shifted and noisy version of the input \( S \).

- The tracking problem:
  Find the best estimation for the signal's position: \( \delta \)

- Solution:
  \( \delta \) corresponds to the position at which the matched filter output is maximal (highest SNR)

- Result:
  \[ \hat{\delta} = \frac{1}{C} \int_{a}^{b} W(\varepsilon) \cdot [y(\varepsilon) - s(\varepsilon)] d\varepsilon \]

  Where:
  \[ \Rightarrow y(\varepsilon) - s(\varepsilon) \] is the difference signal between our new input and our target's last location – this emphasizes the moving part of the signal.

  \[ \Rightarrow W(\varepsilon) = -\frac{\partial s(\varepsilon)}{\partial \varepsilon} \] is the weight function which performs the negative derivative of the signal, i.e., gives the larger weight to the edges in the signal.

  \[ \Rightarrow C = \int_{a}^{b} W^2(\varepsilon) d\varepsilon \] is the energy of the weight function, and is used to normalize the error.

NOTE: By building the weighting function in that way \( W(\varepsilon) = -\frac{\partial s(\varepsilon)}{\partial \varepsilon} \), the weighting function is "matched" to the object's form or shape in the signal. Another choice of \( W \) can provide, as a special cases, the centroid and edges trackers.
4.2 The use of Reference-Map:

Generally, the form of the signal \( s(\varepsilon) \) is not known because of deformation of the object, shadowing and changing aspect angle. Fitts’s approach to that problem was the use of approximation of the signal \( s(\varepsilon) \) (frame, in the 2-D case), which is the exponential smoothing of the past observed signal \( y(\varepsilon) \). In 2-D that leads to the form:

\[
Map(k) = (1 - w)Map(k - 1) + wV(k)
\]

Where:
- \( Map(k) \) is the averaged signal/frame from the beginning of the tracking. (acts like \( s(\varepsilon) \) in 1D)
- \( V(k) \) is the observed (or current) signal/frame. (like \( y(\varepsilon) \) in 1D)
- \( 0 \leq w \leq 1 \) is a scalar weight that can be determined in some ways.

Expression (1) can be also written as:

\[
(2) Map(k) = Map(k - 1) + w(V(k) - Map(k - 1))
\]

Where: \( V(k) - Map(k - 1) \) is the difference signal which emphasizes the moving parts in the frame.

Setting the weight:

In the implementation that will be described in the following chapters the value of the weight is \( w = \max \left\{ \frac{1}{I}, \frac{1}{N-I} \right\} \)

Where:
- \( I \) is the iteration number
- \( N \) is the effective number of frames that have been averaged

Another way is to set \( w \) at the beginning of the tracking to: \( w = \frac{1}{N} \)

Explanation of the frames averaging:

Expression (1) can be written using all the frames until the k'th:

\[
(3) Map(k) = (1 - w)^k Map(0) + wV(k) + w \sum_{j=1}^{k-1} (1 - w)^j V(k - j)
\]

Where: \( Map(0) \) is initialized to the first frame: \( V(0) \)

Submitting \( w = \frac{1}{N} \) to the expression yields:

\[
Map(k) = \left(\frac{N-1}{N}\right)^k Map(0) + \frac{1}{N} V(k) + \frac{1}{N} \sum_{j=1}^{k-1} \left(\frac{N-1}{N}\right)^j V(k - j)
\]
For example, let $N=10$, thus $Map(k)$ is:

$$Map(k) = (0.9)^k Map(0) + 0.1 \cdot V(k) + 0.1 \cdot \sum_{j=1}^{k-1} (0.9)^j V(k - j)$$

If $K=100$, the Reference-Map at the 100'th iteration (frame 100) is:

$$Map(k) = 0.9^{-100} Map(0) + 0.1 \cdot V(100) + 0.1 \cdot \sum_{j=1}^{99} (0.9)^j V(100 - j)$$

It can be clearly seen that the recent frames mostly influence the averaging while the earlier are negligible. Thus, it is said that $N$ is the effective number of frames that have been averaged (in the example: 10 recent frames).

If $V(k)$ will be white circle moving to one direction on a black background then $Map(k)$ after a few frames will look like a commit tail.

**Summarizing the results:**
- Bigger $N$ values $\rightarrow$ more recent frames are effectively averaged
- Smaller $N$ values $\rightarrow$ less recent frames are effectively averaged

Therefore, regarding video tracking, a rule of thumb can be written:
- Slower tracking $\rightarrow$ Bigger $N$ values
- Faster tracking $\rightarrow$ Smaller $N$ values

The averaging of the frames, fundamentally, increases the tracker noise. However, it was found that for $N=8$ the noise increase is negligible.

### 4.3 Mathematical proof of the correlator functions

Assuming Additive White Gaussian Noise model the input signal is:

$$V(x, y) = M(x - \delta_e, y - \delta_d) + n(x, y) \quad \text{assuming we don't "step outside" our frame.}$$

Where:
- $M(x - \delta_e, y - \delta_d)$ $\rightarrow$ the shifted Reference-Map by the correlation errors: $\delta_e, \delta_d$. The correlation errors are represent the target movement in 2D which our correlator had found.
- $V(x, y)$ $\rightarrow$ the observed (or current) scene (frame).
- $n(x, y)$ $\rightarrow$ the AWGN.

Consider the first order Taylor's expansion of $M(x, y)$ around $x - \delta_e, y - \delta_d$:

$$M(x, y) = M(x - \delta_e, y - \delta_d) + \delta_e \frac{\partial}{\partial x} M(x, y) + \delta_d \frac{\partial}{\partial y} M(x, y)$$
Simplifying the notation:

- $\frac{\partial}{\partial x} M(x, y) = G_x(x, y)$
- $\frac{\partial}{\partial y} M(x, y) = G_y(x, y)$

Thus, $M(x - \delta_e, y - \delta_d)$ can be written:

$$M(x - \delta_e, y - \delta_d) = M(x, y) - \delta_e G_x(x, y) - \delta_d G_y(x, y)$$

And the input scene: $V(x, y) \approx M(x, y) - \delta_e G_x(x, y) - \delta_d G_y(x, y) + n(x, y)$

As was mentioned before, the tracker error is such that gives the highest SNR at the output of the Matched Filter. The SNR is defined to be:

$$SNR = \frac{\iint V_k^2(x, y) dxdy}{\iint [V_k(x, y) - M(x - \delta_e, y - \delta_d)]^2 dxdy}$$

The denominator is the Square-Error (SE) of the signal, or the quadratic error, so:

High SNR $\rightarrow$ Low SE of the signal

Basically, the scene and Reference-Map are discrete Matrices. The approximated SE in discrete notation:

$$SE \approx \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \left[ \Delta V_{i,j} + \delta_e G_x(i, j) + \delta_d G_y(i, j) \right]^2$$

where: $\Rightarrow \Delta V_{i,j} = V_{i,j}(k) - M_{i,j}(k-1)$

$\Rightarrow V_{i,j}(k)$ is the k scene

$\Rightarrow M_{i,j}(k-1)$ is the (k-1) Reference-Map

Minimization of SE can be achieved by derivatives in elevation and deflection:

(1) $0 = \frac{\partial SE}{\partial \delta_e} = 2 \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \Delta V_{i,j} \cdot G_x(i, j) + 2\delta_e \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} G_x^2(i, j) + 2\delta_d \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} G_x(i, j) \cdot G_y(i, j)$

(2) $0 = \frac{\partial SE}{\partial \delta_d} = 2 \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \Delta V_{i,j} \cdot G_y(i, j) + 2\delta_d \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} G_y^2(i, j) + 2\delta_e \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} G_x(i, j) \cdot G_y(i, j)$
Using the notation in literature [2]:
\[ W_e(i, j) = G_e(i, j); \quad W_d(i, j) = G_d(i, j) \]
\[ C_e = \sum_{i=1}^{M} \sum_{j=1}^{N} W_e^2(i, j); \quad C_d = \sum_{i=1}^{M} \sum_{j=1}^{N} W_d^2(i, j); \quad C_{ed} = \sum_{i=1}^{M} \sum_{j=1}^{N} W_e(i, j) \cdot W_d(i, j) \]

Thus, equations (1), (2) can be described in matrix form using those notations:
\[ \begin{bmatrix} \sum_{i=1}^{N} \sum_{j=1}^{N} \Delta V_{i,j} \cdot W_e(i, j) \\ \sum_{i=1}^{N} \sum_{j=1}^{N} \Delta V_{i,j} \cdot W_d(i, j) \end{bmatrix} = - \begin{bmatrix} C_e & C_{ed} \\ C_{ed} & C_d \end{bmatrix} \begin{bmatrix} \delta e \\ \delta d \end{bmatrix} \]

The normalized errors are:
\[ \begin{bmatrix} \delta e \\ \delta d \end{bmatrix} = \frac{1}{C_e C_d - C_{ed}^2} \begin{bmatrix} C_d & -C_{ed} \\ -C_{ed} & C_e \end{bmatrix} \begin{bmatrix} \sum_{i=1}^{N} \sum_{j=1}^{N} \Delta V_{i,j} \cdot W_e(i, j) \\ \sum_{i=1}^{N} \sum_{j=1}^{N} \Delta V_{i,j} \cdot W_d(i, j) \end{bmatrix} \]

In the previous paragraph, the 1-D case, the resulting error according to the Matched Filter theory was:
\[ \hat{\delta} = \frac{1}{C_a^b} \int W(\varepsilon) \cdot [y(\varepsilon) - S(\varepsilon)] d\varepsilon \]

This is the raw error of the correlator. In the discrete case, according to Fitts’s notation, the raw errors are:
\[ E_e = \frac{1}{C_e} \sum_{i=1}^{N} \sum_{j=1}^{N} \Delta V_{i,j} \cdot W_e(i, j) \quad \sum_{i=1}^{N} \sum_{j=1}^{N} \Delta V_{i,j} \cdot W_e(i, j) = C_e E_e \]
\[ E_d = \frac{1}{C_d} \sum_{i=1}^{N} \sum_{j=1}^{N} \Delta V_{i,j} \cdot W_d(i, j) \quad \sum_{i=1}^{N} \sum_{j=1}^{N} \Delta V_{i,j} \cdot W_d(i, j) = C_d E_d \]

Substitution of the raw error expressions to the normalized error equations yields:
\[ \begin{bmatrix} \delta e \\ \delta d \end{bmatrix} = \frac{1}{C_e C_d - C_{ed}^2} \begin{bmatrix} C_d & -C_{ed} \\ -C_{ed} & C_e \end{bmatrix} \begin{bmatrix} C_e E_e \\ C_d E_d \end{bmatrix} = \frac{1}{C_e C_d - C_{ed}^2} \begin{bmatrix} C_e E_e - C_d C_{ed} E_d \\ -C_c C_{ed} E_d + C_e C_d E_e \end{bmatrix} \]

Finally, the normalized errors in elevation and deflection:
\[ \delta_e = \frac{1}{1 - \frac{C_{ed}^2}{C_e C_d}} \begin{bmatrix} E_e - C_{ed} E_d \\ C_{ed} E_d \end{bmatrix} \quad \delta_d = \frac{1}{1 - \frac{C_{ed}^2}{C_e C_d}} \begin{bmatrix} E_d - C_{ed} E_e \\ C_{ed} E_e \end{bmatrix} \]
5 Fitts processor algorithm

5.1 The correlator block diagram

5.2 Mathematical discretization of the blocks

5.2.1 Weight matrices

$W_e, W_d$ are the negative derivations of the previous Reference-Map in elevation and deflection accordingly.

Notes:
- Zeros were inserted on the edges of the whole frame in order to avoid problems in the differentials computing.
- The positive directions in the matrix are down for $i$ (elevation) and right for $j$ (deflection).

$$W_e(k)_{(i,j)} = \frac{MAP_{i-1,j}(k-1) - MAP_{i+1,j}(k-1)}{2}$$

$$W_d(k)_{(i,j)} = \frac{MAP_{i,j-1}(k-1) - MAP_{i,j+1}(k-1)}{2}$$
5.2.2 Scale factors

The adaptive scale factors are the summation of the square elements of the weight matrices (i.e., the derivation signal energy):

- \( C_e = \sum_{j=1}^{M} \sum_{k=1}^{N} W_{e,jk}^2 \); where \( M, N \) are the gate size

- \( C_d = \sum_{j=1}^{M} \sum_{k=1}^{N} W_{d,jk}^2 \); where \( M, N \) are the gate size

- \( C_{ed} = \sum_{j=1}^{M} \sum_{k=1}^{N} W_{e,jk} W_{d,jk} \); where \( M, N \) are the gate size

It is used to normalize the raw errors in order to achieve errors in pixel units.

5.2.3 Raw errors

As was described before, the raw errors correspond to the position at which the matched filter output is maximal. That leads to the discrete form of the correlation raw errors:

\[
E_e(k) = \frac{1}{C_e} \sum_{i=1}^{N_e} \sum_{j=1}^{N_e} W_e(k)_{i,j} \times (V_{i,j}(k) - MAP_{i,j}(k-1)) \\
E_d(k) = \frac{1}{C_d} \sum_{i=1}^{N_d} \sum_{j=1}^{N_d} W_d(k)_{i,j} \times (V_{i,j}(k) - MAP_{i,j}(k-1))
\]

Where:

\( \Rightarrow V_{i,j}(k) - MAP_{i,j}(k-1) \) : Emphasize the pixels in which the movement occurs, as we explained before.

\( \Rightarrow W_e(k), W_d(k) \) : Emphasize the edges pixels in elevation and deflection.

5.2.4 Normalized correlation error

\[
\delta_e(k) = \frac{-1}{C_e} \left[ E_e(k) - \frac{C_{ed}}{C_e} E_d(k) \right] \\
\delta_d(k) = \frac{-1}{C_d} \left[ E_d(k) - \frac{C_{ed}}{C_d} E_e(k) \right]
\]
5.2.5 Drift compensation
The drift error compensates for the divergence of the object from the middle of the gated area, and allows accumulation of the total correlation errors. It is computed recursively in each direction, as described below:

\[
\delta^e_{\text{Drift}}(k) = \delta^e_{\text{Drift}}(k-1) + w \times \delta^e(k-1)
\]
\[
\delta^d_{\text{Drift}}(k) = \delta^d_{\text{Drift}}(k-1) + w \times \delta^d(k-1)
\]

As will be seen in the following chapters, the drift correction grows too slow in most cases and doesn't really compensate for the drifting of the object in the gate.

5.2.6 Total correlation error
The total error in the output of the correlator is the sum of the normalized error for the specific iteration and the accumulated drift compensation as described below:

\[
\delta^e_{\text{corr}}(k) = \delta^e(k) + \delta^e_{\text{Drift}}(k)
\]
\[
\delta^d_{\text{corr}}(k) = \delta^d(k) + \delta^d_{\text{Drift}}(k)
\]

This computation includes the delta from the main algorithm flow as well as the drift.

6 Tracking accuracy checking
In order to measure the tracking accuracy and to compare it to the same movie under different tracking method we computed Position-RMSE or RMSE_elevation and RMSE_deflection (RMSE_e and RMSE_d). The units of the RMSE are pixels and it shows the average deviation of the tracker from the tracked object.

RMSE was computed in the following order:
- We prepared with the relevant movie a coordinates array, manually built frame after frame, by using "ginput" Matlab command. One exception was with our first movie of the white Gaussian moving randomly in the frame – since Matlab has the exact coordinates of the Gaussian's center, no manual tracking with "ginput" had to be used.
- After having the manual array we executed the specific algorithm we wanted to check on the movie file and received the tracker coordinates array. We then subtracted the 2 arrays and got the difference array, or the error.
- On the difference array we averaged the values squares and found the root.

Good tracking, besides visually identify it, means low RMSE – lower than half our gate/window sizes in elevation and deflection.

- for the manual array we used the file "ManualTracking"
7 Different implementations and results

We will now examine the different implementations (methods) we used for the algorithm and compare their results. First, every method started with an initialization process:

We received the first frame \( V(0) \), than generated the first Map frame, \( \text{Map}(0) = V(0) \). Then we zeroed all the errors values, received from a known variable the first coordinates of the object and executed the tracker. The block diagram of the basic process is:

![Block diagram of the basic process](image)

Figure 2: basic method flow chart
7.1 "Whole Frame" method

First in our project, we implemented a tracker which works on the entire frame (about 240x240 pixels). All of our matrices and variables are computed on the entire frame.

The block diagram looks like the basic diagram above.

We first tested this method on a synthetic movie we made:

This is a white Gaussian, randomly moves in elevation and deflection, and the tracker (blue cross) is trying to always be on its center. Note that there is absolutely no background. With this movie and method we got good tracking.

RMSE\_elev. = 0.36  \quad RMSE\_defl. = 0.05  \quad (N=8)

RMSE values are smaller than 1 and the tracker kept its relative position on the Gaussian center.
Now we checked what happens when we add shapes→ color→ edges to the background.

To the next movie we added a large square at the corner:

Now the tracker hardly moved!
C variables are larger because of edges matrix W, hence all the errors are smaller – rounded to zero or one pixel.
RMSE_elev. = 4.26   RMSE_defl. = 6.71   (N=8)

RMSE values represent an average distance of the Gaussian center from the tracker, which almost didn't move.
Of course we had to find a way to overcome this problem. "real-life" scenes have much more cluttered background than a white square at the corner. Our next step was to design an option to disregard the background, away from the object.
7.2 "Gate Mask" method

In this method we compute Map and W matrices as usual, on the whole frame. Next we insert our Gate-Mask matrix:

\[
Gate(i, j) = \begin{cases} 
0 & i, j \text{ outside gate area} \\
1 & i, j \text{ inside gate area}
\end{cases}
\]

The normalization and error parameters were computed as:

\[
C_{ed} = \sum_{j=1}^{N_e} \sum_{k=1}^{N_d} W_{ed,j} W_{d,jk} \cdot Gate(i, j) \quad \text{where } N_e, N_d \text{ are the frame size}
\]

\[
C_{e,d} = \sum_{j=1}^{N_e} \sum_{k=1}^{N_d} W_{e,d,j}^2 \cdot Gate(i, j) \quad \text{where } N_e, N_d \text{ are the frame size}
\]

\[
E_{e,d}(k) = \frac{1}{C_{e,d}} \sum_{j=1}^{N_e} \sum_{k=1}^{N_d} W_{e,d,k}(i,j) \cdot \left( V_{i,j}(k) - MAP_{i,j}(k-1) \right) \cdot Gate(i, j)
\]

We actually disregard of what's away from our main object.

We tested this method on 2 movies:
First is the white Gaussian with the white square, to show direct improvement.

We can see the white tracker frame, with the blue cross always in its center, as representing the area of ones in the gate-mask matrix.

This time we got very good results like in our first movie (with no white square):

N=8 , gate size = 60X40
RMSE_elev.=0.42       RMSE_defl. = 0.24
Second is a "real-life" movie of a woman, walking next to a building:

The woman is walking on the sidewalk, enters the shade and exits to the light again just before disappearing in the curve. We can see the tracker frame and the cross at the beginning.
In this movie, with this method, the tracking went well at start but stopped before the shade. One reason is the Map spreading:
Because we capture a whole frame and average it with the current map, the movement of our object causes a deformed Map picture. In this Map picture the background (which is inanimate) is clear. The (moving) target object causes a spread, unclear spot on the sidewalk because of averaging frames with the object at different locations, relatively to the whole frame.
We had to come up with a better gating method than the latter.
7.3 "Gated Frame" method

In this method we are receiving an input frame, cut (gate) it around the target object and computing all the matrices and parameters (Map, W, C, E, etc.) on the small gate. This way we do not get a spread Map and the coming input scene has a similar matrix to relate.

The new block diagram:

Figure 3: "gated frame" flow chart

- Initialization
  - Set LPF coefficient Value: \( w \)
  - Loading the next frame \( V(k) \)
  - Gating the next frame \( V(k) \)
  - Fitts correlation tracker

Generating Reference Map\( (k) \)

Addition of the correlation error to the previous tracker's index in the Map
This movie is the one of the woman on the sidewalk:

![Image of woman on sidewalk](image1)

We managed to track the woman all the way, through the shadow. RMSE values are smaller than half gate size which shows good tracking:

\[
N = 10 \quad \text{; \quad gate size} = 30 \times 10
\]

RMSE_{elev.} = 2.33 \quad \text{RMSE_{defl.} = 1.37}

One can see that RMSE_{e} is a bit larger than RMSE_{d} because the woman movement in the frame is mainly vertical.

The second movie we checked was 'woman and car' movie. The woman, hardly obvious, is walking in almost absolute shadow.

![Image of woman and car](image2)
We got excellent tracking, parallel to the road for the entire movie. 
N=20 ; gate size = 10X10
RMSE_elev. = 1.73      RMSE_defl. = 1.24

This was a difficult movie to work with since the object is very small and hardly noticed.

The third movie is with 2 men, walking on grass, coming in and out from under a tree:

This movie was a good example of hiding. The tracker tracked the object through the first tree until the second tree, slightly to the right.
N=20 ; gate size = 40X10
RMSE_elev. = 6.99      RMSE_defl. = 0.99
Again, we can see that RMSE_e is larger than RMSE_d because of vertical motion. The gate size was chosen because of this reason as well, and because of the object shape.

We also checked our synthetic movie with the white square just for "sanity check" – we got almost the same results as with the "Gate-Mask" and the basic without the square.

N=8, gate size = 60X40

RMSE_elev. = 0.34     RMSE_defl. = 0.32
7.4 Center of Mass addition

We wanted to check a CoM (Center of Mass) addition to the correlation error. We investigated whether it improves our algorithm or not.

The first movie we analyzed this assumption on was one with 2 toy cars on the lab floor:

We got fair tracking for the method with no CoM and excellent tracking for the method with CoM addition.

No center of mass:  
N=8, gate size = 40X30  
RMSE_elev. = 4.23  
RMSE_defl. = 5.07  

![Graph showing index difference over number of frames](image)
RMSE values are 2 times lower with CoM.

Next we tested CoM addition on the movies above and realized, as one can guess, that CoM gives better results only when the object is significantly different (in terms of gray scale) from its close background. The movie with the woman on the white sidewalk became a disaster – the tracker stayed on the brightest spot of the sidewalk. In case we looked for a dark object (CoM can track very bright or very dark spot) the tracker stayed on the bushes on the left or on the shadowed floor.
8 Conclusions and summary

8.1 Conclusions
1. For good tracking we need to use our "Gated Frame" method. This will enable to disregard cluttered background and might even reduce computational load.
2. Once we set the right w (or N) and the right gate, we can achieve excellent tracking through hiding, shadow and other obstacles.
3. CoM addition might help the algorithm when discussing a significantly different object from its close background. In many real-life movies Fitt's tracker will succeed where CoM will fail.

8.2 Summary
We learned in this project a lot about tracking methods. We tested Fitt's algorithm and compared different implementations. Through the project we coded the different methods we checked and tested them on new movies and ones that we made.
The project was interesting and really opened to us the world of tracking algorithms and problems.
We would like to thank Arie Nachmani, our supervisor, and the control lab staff.