A TECHNICAL REPORT

On

IMPLEMENTATION OF A KALMAN FILTER IN POSITIONING FOR AUTONOMOUS VEHICLES, AND ITS SENSITIVITY TO THE PROCESS PARAMETERS

Submitted by

Jeet Patel – 117430479

and

Dhyey Patel – 117396553

For the course

ENPM 667 – CONTROLS FOR ROBOTIC SYSTEMS

Course Instructor –

Dr. WASEEM MALIK

Table Of Contents

List Of Figures

List Of Tables

Abstract

Autonomous vehicles are being used more and more these days in manufacturing industries, transport industry, consumer industry and many more. With these localization and accurate estimation of the positioning of the autonomous vehicle becomes very crucial. Common localization techniques include visual odometry and also localization using triangulation. But with the need of the industry more and more accuracy is required in the positioning of the autonomous vehicle. This technical report analyzes and gives a detail explanation of the journal paper "**Implementation of a Kalman Filter in Positioning for Autonomous Vehicles, and its Sensitivity to the Process Parameters**" and implements the proposed position estimation technique. It is based on accurate position estimation of the autonomous vehicle. The selected journal paper implements the technique of Kalman filter based estimation for accurate position estimation. And the said technique is implemented on an Automated Guided Vehicles (AGV) in a manufacturing industry environment. Finally, the proposed technique in the journal paper has been implemented and its results are discussed.

1 Introduction

1.1 Background On AGVs

To begin with, let us understand what an AGV is. An automated guided vehicle (AGV) or mobile robot is a type of guidance robot system that is not restricted by a fixed range of motion. Rather, it is self-contained and can move along lines, surfaces, or spaces. This is different from a robot arm that is attached to the base and consists of limbs and joints. In most cases, automatic guided vehicles and robot arms are combined. Self-driving cars act as mobile platforms with robotic arms to perform more versatile functions such as remote control, scanning and probing. The AGV is a material flow system or road carrier that autonomously traverses warehouses, distribution centers, or production facilities without onboard operators or drivers.

AGV is used in a variety of applications. It is often used to transport raw materials such as metal, plastic, rubber, and paper. AGVs can, for example, transport raw materials from inbound goods to warehouses or deliver materials directly to production lines. AGVs deliver the raw materials they need continuously and reliably without human intervention, so the materials they need on the production line are always supplied uninterrupted. In addition to transporting raw materials, AGVs are used in work-in-process applications and finished products that support production or assembly lines. In work-in-process applications, AGVs move materials or parts from storage to the production line, or from one workstation to another, enabling repetitive and efficient material transportation throughout the manufacturing process. Without an AGV, a shortage of material on the processing line can stop the manufacturing process. Production is then delayed while human workers remove the required materials from the warehouse and transport them to the production line. The AGV is also used for goods in and out for replenishment and picking. For example, you can use an AGV to move inventory from an in-stock item to a storage location or move it from a long-term storage location of to a picking location to replenish inventory. By moving inventory from long-term storage to forward picking, the picker will have more inventory available and the picking process will be more efficient. AGVs, such as co-mobile robots, support the picking process by guiding warehouse workers on tasks and transporting picked orders to packing and shipping workstations.

Following are the types of AGVs:

- Automatic Guided Carts: An AGC is a simple AGV with limited automation that follows a strip painted on the floor, travels a given path, and performs repetitive actions. AGV carts are widely used in sorting, warehousing, and cross-docking applications to reduce labor costs.
- Unit Loader AGV: These AGV transfer trolleys are designed to move specific parts or products with special tools to move pallets or containers containing multiple items.
- Heavy Load AGVs: These rugged AGVs are used to transport heavy loads over long distances within large systems without the help of fixed route operators. This path can be easily changed by turning or following the floor belt. The

- Forklift/Towing AGVs: This AGV is an automatic guided vehicle developed as a storage robot for warehousing and draws orders from high places using forks and other special tools developed for specific applications.
- Tugger AGV: Tugger AGV uses a trailer filled with products or parts to tow heavy cargo from point A to point B. These automated guided vehicles can make multiple scheduled stops as they move between factories, factories, or warehouses.

1.2 Background On Position Estimation Techniques

Now let us understand what localization is and how it is crucial for an AGV. Localization or positioning is a core component of an autonomous guided vehicle. Since, AGV drives on their own it is constantly required to know the current position of the AGV in order to reach the desired location or in order to carry out desired task. Positioning or localization is a capability of an AGV which makes them pinpoint to their current location inside the space. Various methods have been evolved throughout past few years. And the accuracy of the positioning system kept on increasing after evolution of every new technique. Selection of a particular positioning technique largely depends on the kind of application and kind of the resources like sensors and other stuff. Let us discuss some commonly used positioning techniques in AGVs.

The following are the commonly used position estimation techniques:

- **Free Range positioning technique:** It is very flexible and widely used in autonomous vehicles. Since, it is very good for unstructured and unknown environments it is very popular technique.
- **Rigid navigation System**: Rigid navigation systems guide vehicles with limited flexibility in predefined lanes. Common fixing methods include mechanical railroad tracks which is similar to railroad tracks, electrical / magnetic systems that guide vehicles through underfloor wire guides, optical / chemical strips placed on the surface of the floor.
- **Odometry**: Odometry is widely used to evaluate the position and orientation of mobile robots. This is one of the most common ways to evaluate relative position. In most mobile robots, the offset from a known starting position is calculated by an encoder mounted on the wheel to track wheel speed and/or steering angle. robot wheels. Odometry can be realized independently without external information. Therefore, odometer measurement is simple, inexpensive, and easily performed in real time. However, odometry is greatly influenced by internal systematic and non-systematic factors, and errors accumulate infinitely. Therefore, correction of odometry errors is very important for accurate odometry.
- **Dead Reckoning System**: Dead reckoning is an incremental positioning method that allows you to estimate the total distance covered from the starting point. Dead reckoning systems do not require external sensor information to estimate position and orientation. However, dead reckoning systems have the problem that estimation errors accumulate over time. Therefore, these methods require regular updates to correct any errors that accumulate in the robot's position.

- **Inertial based Navigation System**: The Inertial Navigation System is a self-contained device consisting of an Inertial Measurement Unit (IMU) and a Computational Unit. An IMU typically consists of a 3-axis accelerometer, a 3-axis gyroscope, and possibly a 3-axis magnetometer, which measures the angular velocity and acceleration of the system. A computing unit used to determine the attitude, position, and speed of a system based on raw measurements from an IMU given an initial starting position and orientation. So, basically INS uses a computer, motion sensor (accelerometer), and rotation sensor (gyroscope) to position and direct moving objects without the need for external references and a navigation device that continuously calculates speed ,direction and speed of movement.
- **Triangulation**: Triangulation is a method of calculating a position based on a known distance between two measuring devices and the measurement angle from these two points to the object. This works in conjunction with the angle-side-angle triangle congruence theorem to find the position of an object.
- **Beacon Based Navigation**: In this method the positioning of the AGV is carried out by the relative positioning of the vehicle with respect to the beacons or landmarks whose position is already known. The measurements can be angular as well as linear. And the beacons can be any transmitter receiver system like ultrasonic sensors, satellites, or infrared waves.

Moreover, it is not appropriate to rely on a single technique of positioning if very high accuracy is required. In this case fusion comes in the role. Fusing the data and outputs of various sensor and then estimating the desired result is known as sensor fusion. Even in the task of positioning sensor fusion improves the accuracy very drastically. Estimation of position often uses different data like odometry with other sensors to get the correct estimation.

In the selected journal paper a robust positioning system has been proposed. The proposed positioning system fuses the data of odometry, a rotating beam of lasers having retro reflectors and Kalman filter. And the proposed positioning technique is tested on a mobile robot ROBI, which is placed in an industrial environment. The industrial environment is a flexible manufacturing system. The system includes two CNC cells with robots for loading and unloading, two robot assembly stations, an inspection and quality control station, and a linear one-way conveyor for material transfer. Each CNC cell and conveyor belt has a docking point for interfacing with the AGV.

2 Overview Of The Mobile Robot ROBI Used In The Selected Journal Paper

This section discusses the structure of the mobile robot ROBI which is used for testing. In the selected journal paper the proposed positioning technique is tested on a mobile robot ROBI. This mobile robot is shown in Figure 1. The type of the driving unit is asynchronous. So, basically there are two types synchronous and asynchronous motors. Also there are different types of motors such as servo motors which are rotary or linear drives that allow precise control of angular or linear positions, velocities, and accelerations. It consists of a suitable motor coupled with a position

feedback sensor. It also requires a relatively high degree of control and often requires a dedicated module designed specifically for servomotors. Also, one commonly used motor is DC brushless motor, which was first developed to deliver superior performance in less space than a brushed DC motor. These motors are smaller than the AC model. The controller is built into the electric motor, which simplifies the process without commutators or slip rings. And in the mobile robot ROBI two stepper motors have been fitted. The stepper motors are used instead of stable rotation to provide step angle rotation. The total angle of rotation of each rotor is known to be 180 degrees. However, you can use a stepper motor to divide the total rotation angle into many steps, such as 10 degrees x 18 steps. This means that the rotor will gradually move 18 times, 10 degrees each time, in a complete cycle of rotation. This is very useful for the given task of material handling. Apart from that the mobile robot has four castor wheels. There are generally three types of wheels:

- **Standard Wheel**: Rotation around the wheel axle and rotation around the contact point are the two degrees of freedom available on a conventional wheel. With respect to the contact point, the axis of rotation runs through the center of the wheel, usually along the plane of the wheel. If the wheels are aligned with no camber and toe, this design permits steering without adding additional pressures to the robot chassis.
- **Castor Wheel**: A caster wheel, like a conventional wheel, has two degrees of freedom. One revolves around the wheel axis, while the other revolves around an offset from the wheel's center. Caster wheels are commonly utilized to give chassis support. Because steering using caster wheels generates stresses on the chassis, it is rarely employed for navigating and delivering motion. The primary benefit of caster wheels is that they automatically align when moving forward after turning.
- **Mecanum Wheel**: A Mecanum wheel, also known as a Swedish wheel, has three layers of freedom: rotation across the wheel axis, rotation across the rollers, and rotation across the touch point. A Mecanum wheel features rollers arranged at 45° angles around the circle of the primary wheel. An omnidirectional wheel with rollers oriented at 90° angles is another example of this type of wheel. Three or more Mecanum wheels are mounted to the chassis and circled in a mixture of clockwise and counterclockwise rotations to provide omnidirectional movement.

And every motor is controlled by a respective controller which also provides pulse frequency. The linear velocity equations are given by,

$$V_{\text{left}} = \frac{2\pi R_{\text{left}}\, n_{\text{left}}}{NT}$$

$$V_{\text{right}} = \frac{2\pi R_{\text{right}}\, n_{\text{right}}}{NT}$$

where, $\boldsymbol{R_{\text{left}}}$ and $\boldsymbol{R_{\text{right}}}$ are the radii of the right and left wheels,

T is the gear ratio from the driving motor to the wheels,

N is the motor resolution (pulse per revolution), and

$\boldsymbol{n}_{\mathbf{left}}$ and $\boldsymbol{n}_{\mathbf{right}}$ are the pulse frequencies for the left and right wheels.

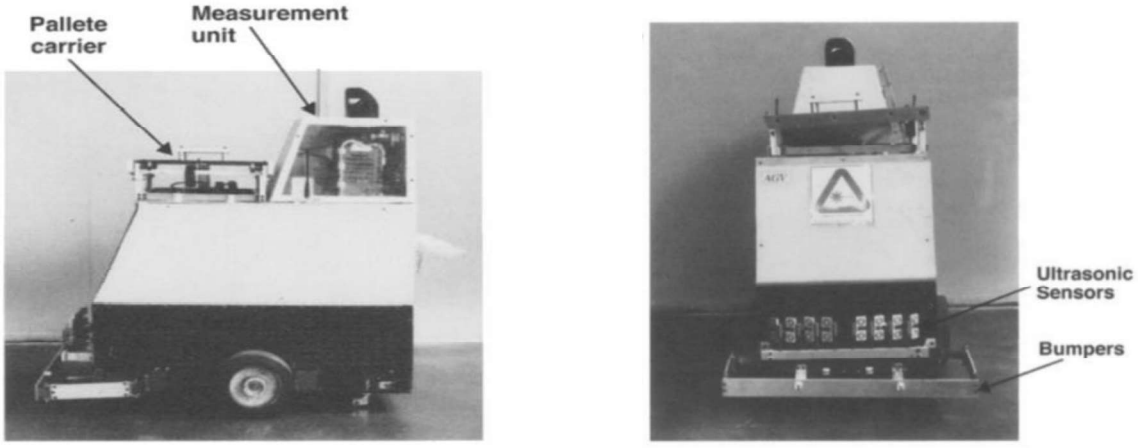

figure 1 Front and side view of mobile robot ROBI [1]

Moreover, the measurement unit consists of a laser system which is low powered. A rotating laser is a more advanced laser that rotates a light beam fast enough to produce a 360° horizontal or vertical effect, illuminating not only the solid line but also the horizontal plane. Laser beam projectors use a rotating head with a mirror to rotate the laser beam around a vertical axis. If the mirror is not self-leveling, it is equipped with a human-readable level and manually adjustable screws to level the projector. Personnel carried by the operator are equipped with a movable sensor that can detect the laser beam and output a signal when the sensor is aligned with the beam.

The measuring unit shown in figure 2 consists of a rotating low power laser system. The system continuously sends a thin beam parallel to the ground. The plurality of reflectors are arranged at predetermined positions in the working area in order to reflect the laser beam back to the photodetector on the robot. The angular position of the rotating assembly is measured by the optical encoder, which detects the position as soon as the optical sensor detects the reflected beam. A series of optical lenses and mirrors focus the transmitted and reflected laser beam to the required wavelengths. The device rotates at 3 Hz and can reliably detect reflectors up to 10 m (30 feet) from a vehicle. The system resolution is 1.7 x 10.3 rad (0.1 °). The reflector is 500m long and has a square profile 15 × 15mm. In theory, a circular profile is preferred, but such a profile reduces the bandwidth of the reflector and reduces the effective distance to only 3 m (10 ft).

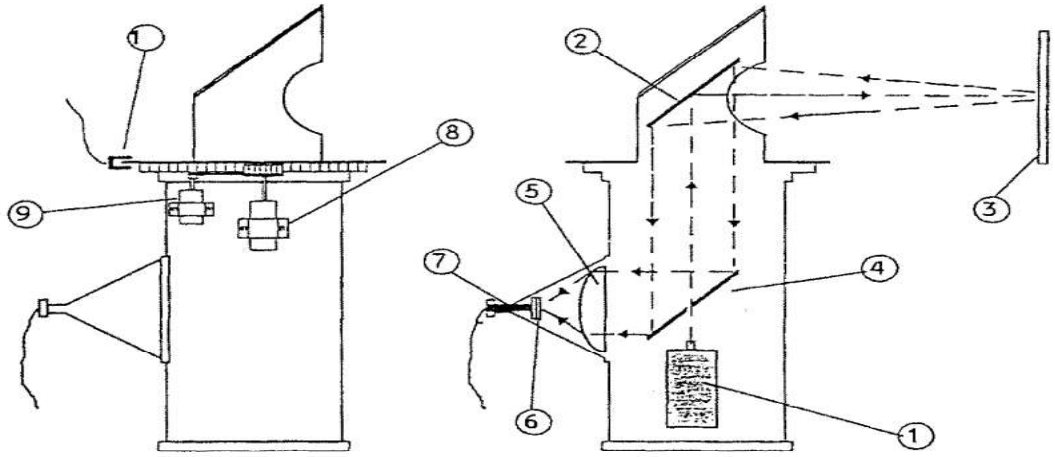

Figure 2 The measuring unit. 1. Laser transmitter. 2. Rotating mirror. 3. Reflector. 4. Stationary mirror. 5. Lens. 6. Optical filter. 7. Optical sensor. 8. D.C. Motor. 9. Encoder. 10. Calibration sensor.[1]

Figure 3 shows a vehicle entering one of the conveyors docking stations. The loading operation raises the lift before the AGV enters the port. After accurate positioning, lower the lift and pull the pallet out of the conveyor belt. For unloading, the process is reversed and the AGV moves to the port with the lift in the lower position. After the vehicle is placed, the lift lifts the pallet and the vehicle leaves the harbor. The robot is equipped with a mechanical bumper that cuts off the power supply to the drive motor in the event of a collision. It also has an ultrasonic sensor, communication antenna and control / interface panel for obstacle detection and avoidance.

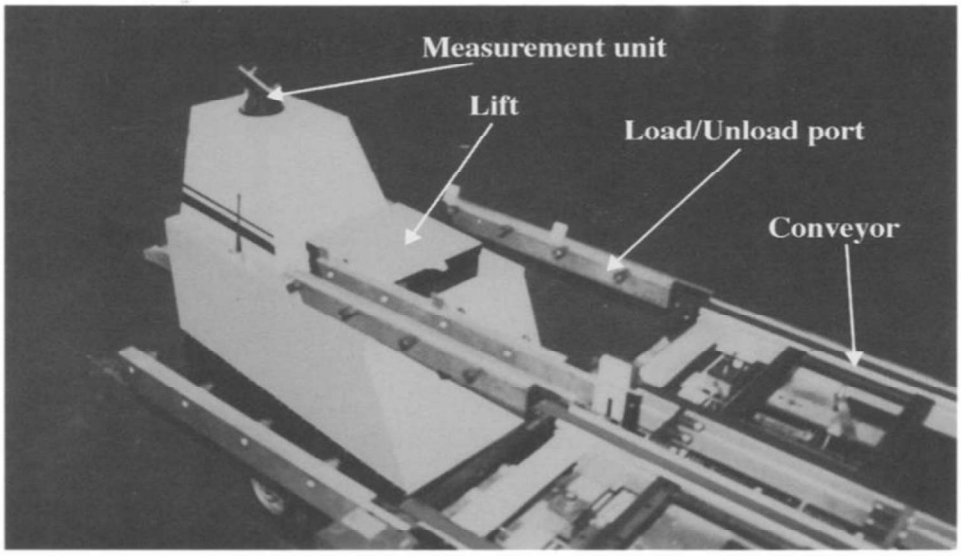

Fig 3 Vehicle entering the docking station [1]

3 Kalman Filter And It's Overview

3.1 Introduction

This section gives brief introduction on Kalman filter. One of the most significant and widely used estimate algorithms is the Kalman Filter. The Kalman Filter generates hidden variable estimates based on erroneous and imprecise measurements. In addition, the Kalman Filter predicts the future state of the system based on previous estimates.

In the face of uncertainty, one of the most difficult issues for tracking and control systems is to provide accurate and precise assessment of hidden variables. For example, consider the measurement uncertainty of GPS receivers is affected by a variety of external factors, including thermal noise, atmospheric effects, minor variations in satellite locations, receiver clock precision, and so on. In such case Kalman filters prove to be very useful and crucial for the accurate estimation.

Let's understand the need of prediction. Consider a radar based position tracking system:

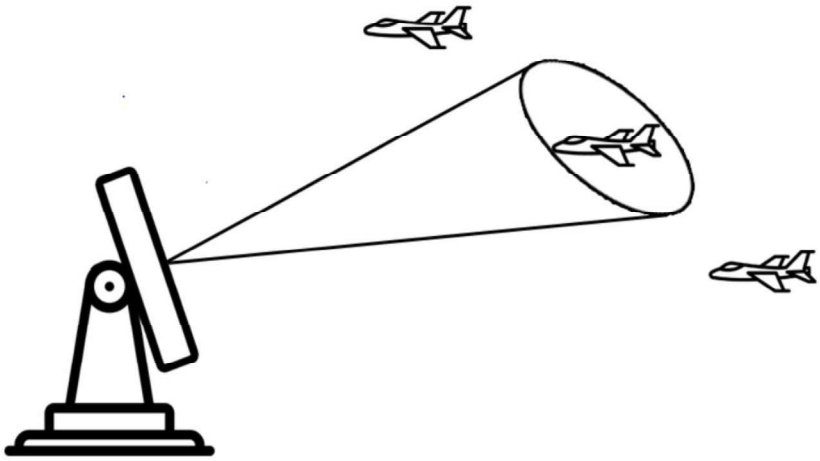

Fig4 Radar based position tracking system [1]

A pencil beam is sent in the direction of the target by the tracking radar. Assume a 5 second track cycle. In other words, the radar returns to the target every 5 seconds by emitting a specific track beam in its direction. The radar assesses the current target position and velocity after sending the beam. In addition, the radar anticipates (or estimates) the target position at the next track beam.

The future target location can be simply determined using Newton's motion equations:

$$x = x_0 + v_0\Delta t + \frac{1}{2}a\Delta t^2$$

Where, x = target's position

x_0 = target's initial position

v_0 = target's initial velocity

a = target's acceleration

Δt = time interval (5 seconds in the example)

Newton's motion equations in 3 dimensions can be written as:

$$x = x_0 + v_{x0}\Delta t + \frac{1}{2}a_x\Delta t^2$$

$$y = y_0 + v_{y0}\Delta t + \frac{1}{2}a_y\Delta t^2$$

$$z = z_0 + v_{z0}\Delta t + \frac{1}{2}a_z\Delta t^2$$

A System State is made up of the target parameters [x , y, z, v_x, v_y, v_z, a_x, a_y, a_z]. The prediction algorithm takes the current state as input and outputs the next state (the target parameters for the following time period). A Dynamic Model is the collection of equations described above. The input-output relationship is described by the Dynamic Model.

Let's go back to our example. As we can see, if we know the current state and the dynamic model, we can simply anticipate the next target state. They aren't, to be sure. To begin with, the radar measurement is not exact. It has a random error in it or uncertainty. The extent of the inaccuracy is determined by a number of factors, including radar calibration, beam width, and return echo magnitude.

Measurement noise refers to the inaccuracy that is incorporated in the measurement. Furthermore, due to external influences such as wind, air turbulence, pilot movements, and so on, the intended motion is not exactly aligned to motion equations. Process Noise refers to the dynamic model mistake. The projected target location may differ significantly from the real target position due to Measurement and Process Noise. The radar may deliver the track beam in the wrong direction and miss the target in this instance. We need a prediction algorithm that takes process and measurement uncertainty into account in order to increase radar tracking performance. And the most widely used prediction algorithm is Kalman filter.

Kalman filters are used to estimate states in state space format based on linear dynamical systems. The state evolution from time k-1 to time k is defined by the process model as follows:

$$\boldsymbol{x}_k = F\boldsymbol{x}_{k-1} + B\boldsymbol{u}_{k-1} + \boldsymbol{w}_{k-1}$$

where, F is the state transition matrix applied to the previous state vector $\boldsymbol{x}_{k-1}$,

B is the control-input matrix applied to the control vector $\boldsymbol{u}_{k-1}$,

$\boldsymbol{w}_{k-1}$ is the process noise vector that is assumed to be zero-mean Gaussian with the covariance Q, i.e. $\boldsymbol{w}_{k-1} \sim \mathcal{N}(0, Q)$

The measurement model is associated with the process model, which is and represents the relationship between the state and the measurement at the current time step k,

$$z_k = H\boldsymbol{x}_k + \boldsymbol{v}_k$$

where z_k is the measurement vector, H is the measurement matrix, and $\boldsymbol{v}_k$ is the measurement noise vector that is assumed to be zero-mean Gaussian with the covariance R, i.e., $\boldsymbol{v}_k \sim \mathcal{N}(0, R)$.

The Kalman filter's job is to generate an estimate of $\boldsymbol{x}_k$ at time k, given the initial estimate of $\boldsymbol{x}_0$, the series of measurement, $z_1, z_2, \ldots, z_k$, and the system information provided by F, B, H, Q, and R. Subscripts to these matrices are removed here since they are assumed to be invariant across time, as they are in most applications. In many actual applications, the true statistics of the noises are not known or are not Gaussian, even though the covariance matrices are supposed to reflect them. As a result, Q and R are frequently employed as tuning factors that the user can tweak to achieve the desired result.

3.2 Kalman Filter Equations

The equations shown in table 1 are the five main equations of kalman filter.

Equation	Equation Name
$\hat{x}_{n,n} = \hat{x}_{n,n-1} + K_n\left(z_n - \hat{x}_{n,n-1}\right)$	State update
$\hat{x}_{n+1,n} = \hat{x}_{n,n} + \Delta t\hat{\dot{x}}_{n,n}$ $\hat{\dot{x}}_{n+1,n} = \hat{\dot{x}}_{n,n}$ (for constant velocity dynamics)	State Exploration
$K_n = \frac{p_{n,n-1}}{p_{n,n-1} + r_n}$	Kalman Gain
$p_{n,n} = (1 - K_n)p_{n,n-1}$	Covariance update
$p_{n+1,n} = p_{n,n}$	Covariance Extrapolation

Table 1 Kalman filter equations

- State Update Equation:

- $\hat{x}_{n,n} = \hat{x}_{n,n-1} + K_n\left(z_n - \hat{x}_{n,n-1}\right)$
- The measuring residual, often known as the innovation, is denoted by the phrase $\left(z_n - \hat{x}_{n,n-1}\right)$. The new knowledge is contained in the innovation.
- As n increases, (1-n) decreases in our example. This indicates that we don't have enough information about the current state to estimate it at first, therefore we rely on measurements. Because (1-n) diminishes as we go on, each subsequent measurement has less weight in the estimating process. The new measurements' contribution would eventually become insignificant. It will be called initial guess and it will be the first estimate.
- K_n is known as the Kalman gain factor.

- State Extrapolation Equation(also known as Prediction Equation):
  - Assume that the world is one-dimensional. We'll use the example of a robot moving away from the radar in a radial direction (or towards the radar). The angle to the radar is constant in a one-dimensional universe.
  - The range to the robot at time n is represented by $\hat{x}_{n,n}$. The rate of change of the range with respect to time is defined as the robot velocity, and hence the velocity is a derivative of the range :

$$\dot{x} = v = \frac{dx}{dt}$$

  - The radar sends a steady track beam in the direction of the target. t is the track-to-track interval.
  - The dynamic model of the system can be characterized by two equations of motion, assuming constant velocity, $\hat{x}_{n+1,n} = \hat{x}_{n,n} + \Delta t\hat{\dot{x}}_{n,n}$ and $\hat{\dot{x}}_{n+1,n} = \hat{\dot{x}}_{n,n}$.
  - The robot range at the next track cycle equals the current track cycle plus the target velocity multiplied by the track-to-track gap, according to these formulae. Because we're assuming constant velocity in this case, the next cycle's velocity matches the current cycle's velocity.
  - The above equation system is referred to as a State Extrapolation Equation (also known as a Transition Equation or a Prediction Equation) and is one of the five Kalman filter equations. The present condition is extrapolated to the next state using this system of equations (prediction).

- Kalman Gain Equation:
  - The Kalman Gain (K_n) is the weight that we assign to the measurement, as you can see. And (1-K_n) is the value we assign to the estimate.
  - The Kalman Gain is a value that ranges from 0 to 1: $0 \leq K_n \leq 1$.
  - The Kalman Gain is near to zero when the measurement uncertainty is big and the estimate uncertainty is modest. As a result, we assign the estimate a high weight and the measurement a low weight. The Kalman Gain, on the other hand, is near to one when the measurement uncertainty is small and the estimate uncertainty is significant.

    - As a result, we give the estimate a low weight and the measurement a high weight. The Kalman gain equals 0.5 when the measurement uncertainty equals the estimate uncertainty.
    - The α -β (-γ) in a Kalman filter are calculated dynamically for each filter iteration. These values are known as Kalman Gain and are denoted by K_n.

- Covariance Update Equation:
    - This equation updates the current state's estimated uncertainty. The Covariance Update Equation is what it's called.
    - Since (1-K_n)<=1, it is obvious from the equation that the estimate uncertainty gets less with each filter iteration. When the measurement uncertainty is high, the Kalman gain is low, resulting in a sluggish convergence of the estimate uncertainty. When the measurement uncertainty is minimal, however, the Kalman gain is substantial, and the estimate uncertainty quickly approaches zero.

- Covariance Extrapolation Equation:
    - The covariance extrapolation equation is also known as the estimate uncertainty equation. It estimates the required state.
    - For example consider radar based positioning. The current location estimate uncertainty plus the current velocity estimate uncertainty multiplied by time squared equals the predicted position estimate uncertainty. The uncertainty of the expected velocity estimate is the same as the uncertainty of the present velocity estimate.

3.3 Summary

The Kalman filter cycle can be summarized as follows:

- Step 1: The initialization is performed only once.
    - Initial state system.
    - Initial state uncertainty.
    - Initialization is followed by prediction.
- Step1: Measurement
    - Measurement state system.
    - Measurement uncertainty.
- State2: State Update
    - The state update process is responsible for current state estimation.
    - Measured value , measurement uncertainty.
    - Previous system state estimate and estimate uncertainty.
    - On the basis of inputs the state update process will give the Kalman gain and will provide current system state estimate and also current state estimate uncertainty.
- State3: Prediction

- Based on the system's dynamic model, the prediction process extrapolates the present system state and the uncertainty of the current system state estimate to the next system state.
- The initialization results are treated as the Previous State Estimate and Uncertainty in the first filter iteration.
- The prediction outputs become the Previous State Estimate and Uncertainty as the filter iterations go.

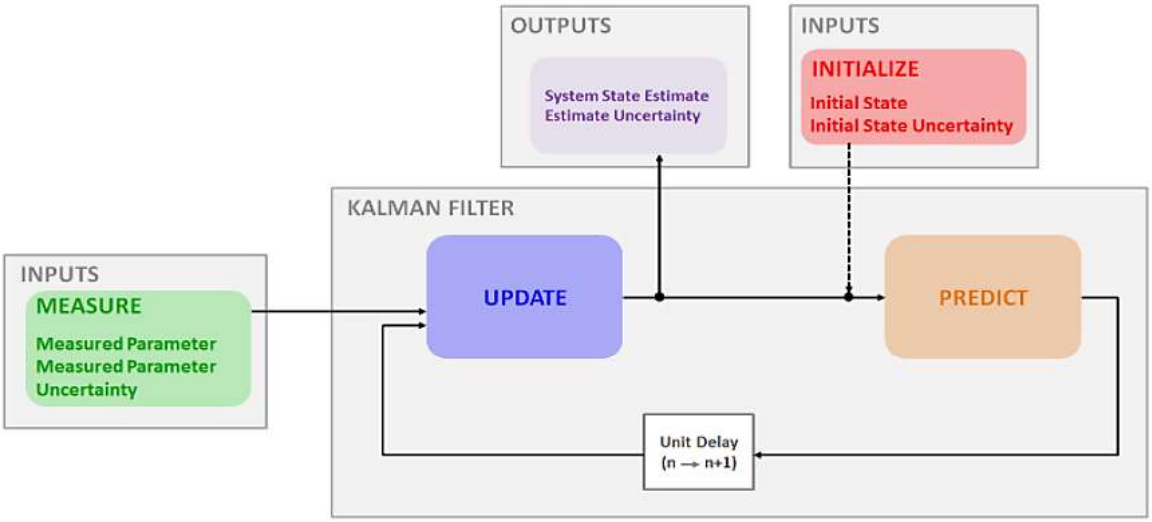

Fig 5 Block diagram of Kalman filter cycle

4 Position Estimation with Kalman Filter

This section discusses the position estimation based on Kalman filter which is implemented in the selected journal paper. For the mobile robot in the selected journal paper, there are two sources of data for the estimation of the position of the mobile robot. The first one is the data from Odometry. And the second is the data from triangulation. But none of the mentioned techniques are sufficient for the accurate estimation of the position of the mobile robot. This is because, only relative position can be determined using odometry and it starts accumulating errors as the distance travelled increases. Furthermore, accurate positioning of the robot can be obtained by using method of triangulation if the robot is stationary but the robot is not going to be stationary always. And if triangulation method is performed while the robot is in motion then very inaccurate positioning estimate can be obtained. All these factors lead to having some method which is more reliable and accurate. As a result a need for fusion arises. In the method used in the journal paper, the data of odometry and the angular measurements of the beacons can be fused with the Kalman filter which gives the accurate estimation of the position of the mobile robot.

The following shows the parameters of the filter:

- $\mathbf{X}(k) = (x_k, y_k, \theta_k)^T$, which shows the lateral position x, y and orientation θ at discrete time k.
- γ_j denotes the measurement of the j th landmark $\hat{\mathbf{X}}(k+1 \mid k)$. denotes the prediction of the robot pose at discrete time k+1 based on the pose at time k.
- $\tilde{\gamma}_j(k+1 \mid k)$, denotes the prediction of the angular measurement to the j th reflector at the discrete time k+1 with respect to the pose at time k.
- $\hat{\mathbf{X}}(k+1 \mid k)$ gives the estimate of robot pose at discrete time k+1.

There are two modules of the Kalman filter used for the estimation of the position of the robot.

4.1 State Prediction:

The state prediction is based on the motion model of the robot. Consider a vector $\mathbf{u}_k$ as:

$$\mathbf{u}_k = \begin{pmatrix} \delta d_k \\ \delta \theta_k \end{pmatrix}$$

Where,

$$\delta d_k = v_k Ts$$

And

$$\delta \theta_k = \frac{v_k}{p_k} Ts$$

$$T_s \; V_k \; \rho_k$$

Here in the above equations, T_s represents the sampling rate and the linear velocity of the robot center is represented by V_k. Also, the momentary radius of curvature of the path of the vehicle is denoted by ρ_k . The equations of V_k and ρ_k is given by following:

$$V_k = \frac{V_{\text{left}} + V_{\text{right}}}{2}$$

$$p_k = \frac{D\left(V_{\text{right}} + V_{\text{left}}\right)}{2\left(V_{\text{right}} - V_{\text{left}}\right)}$$

Here in the above equations V_{left} and V_{right} are determined by the equation's above. Also, D is the distance between the two driving wheels.

For odometry models, for small angles the f is denoted by:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{u}_k(1)\cos\left(\theta_k + \frac{1}{2}(\mathbf{u}_k(2))\right.$$

$$y_{k+1} = y_k + \mathbf{u}_k(1)\sin\left(\theta_k + \frac{1}{2}\left(\mathbf{u}_k(2)\right)\right.$$

$$\theta_{k+1} = \theta_k + \mathbf{u}_k(2)$$

4.2 Measurement Prediction

Considering a robot state at a discrete time $(k+1), \mathbf{X}_{k+1} = (x_{k+1}, y_{k+1}, \vartheta_{k+1})^T$, here the γ_j represents the angle between the vector of direction of the robot u, which is a unit vector and the vector of the reflected beam t. Here the vector of the reflected beam t is the vector difference of the vector pointing to the center of the receiver or transmitter r and the beacon vector $\mathbf{R}_j$. The following equations represents the measurement parameters:

$$\mathbf{u} = (\cos\ \theta_{k+1}, \sin\ \theta_{k+1})^T$$

$$|\gamma_\lambda| = a\cos\left(\frac{\mathbf{u}\cdot\mathbf{t}}{|\mathbf{u}|\cdot|\mathbf{t}|}\right)$$

$$|\gamma_t| = a\sin\left(\frac{|\mathbf{u}\times\mathbf{t}|}{|\mathbf{u}|\cdot|\mathbf{t}|}\right)$$

$$\gamma_j = \text{atan}\ 2(|\mathbf{u}\times\mathbf{t}|, \mathbf{u}.\mathbf{t})$$

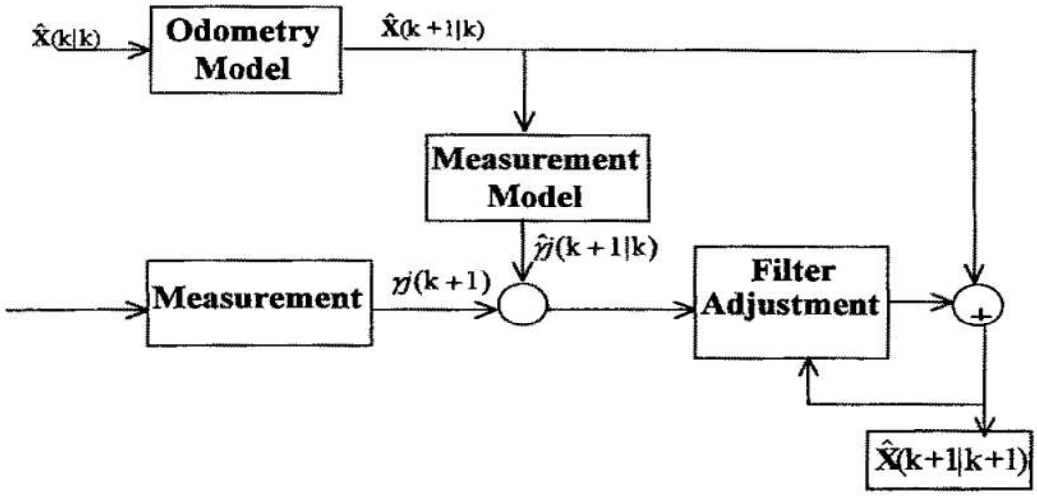

Fig6 Block diagram of position estimation algorithm proposed in the journal paper [1]

$$\mathbf{t} = \mathbf{R}_j - \mathbf{r}$$

$$\gamma_j = \text{atan}\ 2\left(|\mathbf{ux}(R_j - \mathbf{r})|, \mathbf{u}\cdot(\mathbf{R}_j - \mathbf{r})\right)$$

In case of offset between the robot center and the laser unit the r is determined by the equation

$$\mathbf{r} = \mathbf{X}' + \mathbf{c}$$

Where,

$$\mathbf{X}' = (\boldsymbol{x}_{k+1}, y_{k+1})^T \text{ and } \mathbf{c} = -L\mathbf{u}.$$

Here L is the offset between the laser unit and the center of the mobile robot ROBI. Finally, the equations of the measuring functions are represented as follows:

$$\hat{\gamma}_j(k+1 \mid k) = a\tan 2\left(\left|\mathbf{u}x\left(\mathbf{R}_j - \mathbf{X}'\right)\right|, \mathbf{u} \cdot \left(\mathbf{R}_j - \mathbf{X}'\right) + L\right)$$

$$\hat{\gamma}_j(k+1 \mid k) = h_j(\widehat{\mathbf{X}}(k+1 \mid k))$$

4.3 Kalman Gain Selection

Kalman filter gain is one of the most crucial parts of Kalman filter. The performance of Kalman filter depends on the value of the Kalman filter gain. The value of Kalman gain is between 0 and 1. When producing a new estimate, the Kalman Gain defines a weight for the measurement and a weight for the old estimate. A significant Kalman Gain would arise from a low measurement uncertainty compared to the estimate uncertainty (close to 1). This indicates that the new estimate will be reasonably close to the measurement. A low Kalman Gain would arise from a high measurement uncertainty compared to the estimate uncertainty (close to 0). This implies that the new estimate will be similar to the prior one.

There are several methods to determine the gain of Kalman filter. Here, the Kalman filter's gain is derived by statistically analyzing the models and measurements occurring during the process. This statistical analysis provides a covariance matrix which as a result gives the Kalman filter gain.

To begin with, first of all it is required to consider the different errors which may occur considering the odometry model of the mobile robot. The sources of errors might be like the slippage occurring in the wheels or because of the uneven floor. Errors may also arise because of the uncertainty in the parameters related to the mechanical aspect of the mobile robot like, wheel alignment, wheels diameter, distance between the wheels and some other sources.

Considering the numerical errors which may occur in geometric calculations[4] suggested to random error in geometric calculation of both the right and left wheel. This was termed as adding white noise and each of the random noise was independent of each other. And based on this the gain is determined by experiment. Also, considering a new input vector which also includes the errors and also considering the actual angular and lateral displacements with errors the distance D which is the distance between both the wheels can be determined. And subsequently, odometry model can described as a combination of vector function and coefficient matrix of the error vector. And finally, the process covariance matrix can be obtained.

5 Results (From Tests Performed in The Journal Paper)

This section discusses about the results derived from the test and simulation performed on the positioning estimation technique described in the journal paper. In total three different experiments with totally different scenarios and conditions were conducted. Each experiment will be discussed in the section.

The first experiment was conducted considering the most basic and the ideal case. In the first experiment the simulation was carried out for the motion of the mobile robot ROBI on a straight path which was 8metres long. Following are conditions of the experiment:

- The average travel speed was set at 0.6 m/s.
- The number of reflectors which were used were 8.
- The frequency of the measurement was 5Hz.

The offset of the odometry model compared against the offset obtained from the Kalman filter is shown in the graph of figure 7. The following conclusions can be drawn from the simulation results:

- As time and distance increases the offset of the odometry also increases.
- The offset derived from the Kalman filter was always less than 1cm.
- The error of the model of odometry estimation model gives a total lateral error of more than 0.2metres and in terms of orientation its gives error of 0.07 rad.
- On the other hand, the error in the estimation of lateral position is less than 1cm and the error in orientation is less than 0.02 rad.

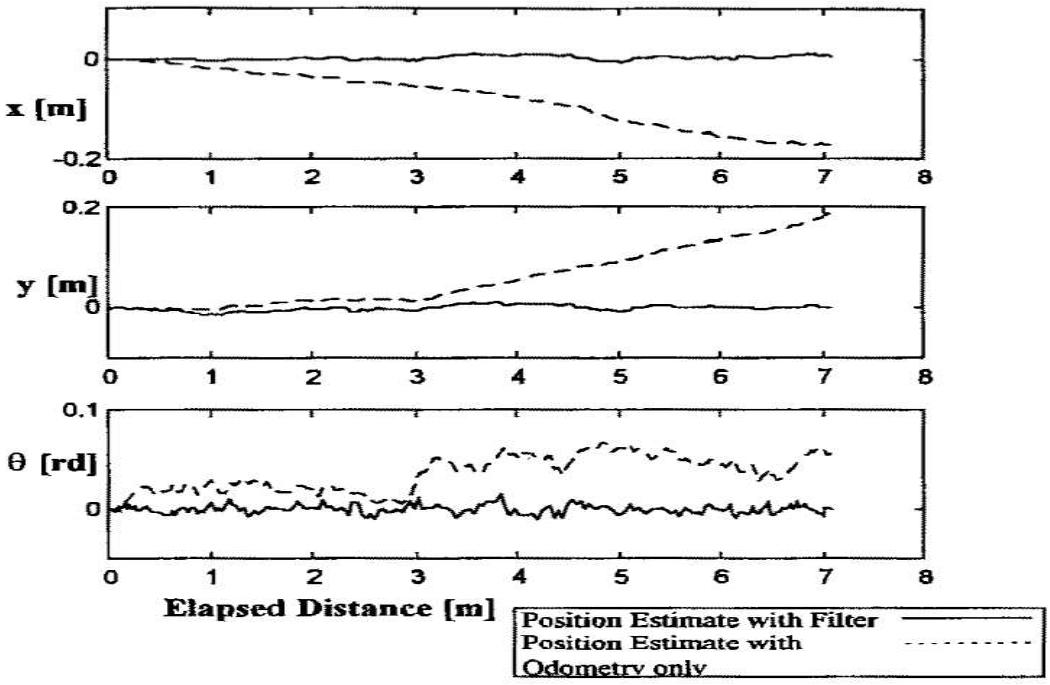

Fig7 The position estimate of the Odometry model compared with the offset determined by the Kalman filter[1]

In the second experiment the conditions were made more practical. And as a part of that disturbances were introduced in each of the wheels. The disturbances were introduced in each wheel for 0.2 seconds. And the following conclusions were drawn from the experiment:

- The error of the model of odometry estimation model gives a total lateral error of more than 0.4metres and in terms of orientation its gives error of 0.05 rad.
- Whereas the Kalman filter significantly reduces both the errors to their minimum value after about 2 seconds.

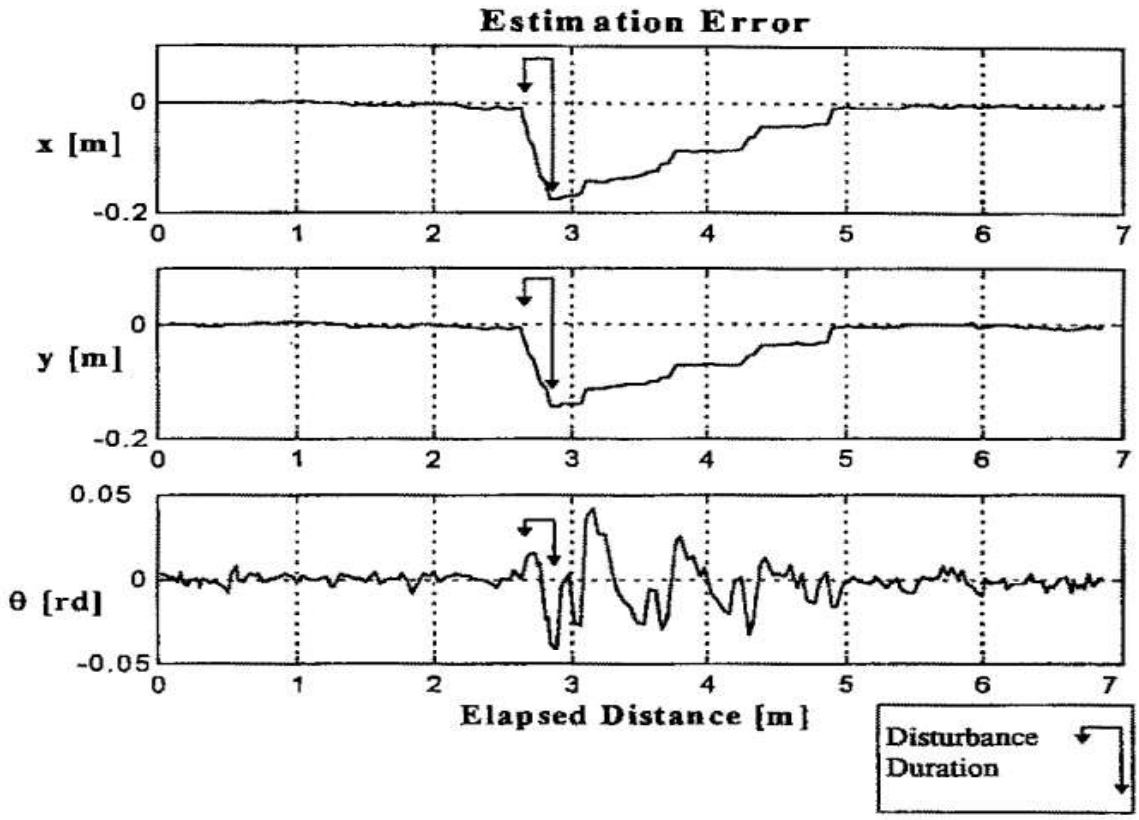

Figure 8 . Effects of introducing an external disturbance[1]

In the next experiment the testing conditions were made stricter. And the response of the positioning estimation system was tested against the introduction of initial error in the position estimation. In the final experiment there are two parts. In the first part an initial offset of 10cm was introduced in the system before testing started. And following conclusions were drawn from the results:

- Damping of the error takes place.
- And the position estimate reaches its nominal value after 3 seconds.

- Also, some crucial phenomenon was noted. Regardless of the error introduced in the lateral position of the mobile robot, even the angular estimation was affected.
- This phenomenon occurred because of the Kalman filter. The Kalman filter first of all compares the angular measurement of the reflectors with actual measurement.
- But since, initially a large deviation was detected by the Kalman filter because of the initial offset and the cause of the deviation can be related to either angular error or lateral error.
- Finally, the source of the error is detected by the Kalman filter after 1 seconds and the error in angular estimate is drastically reduced.

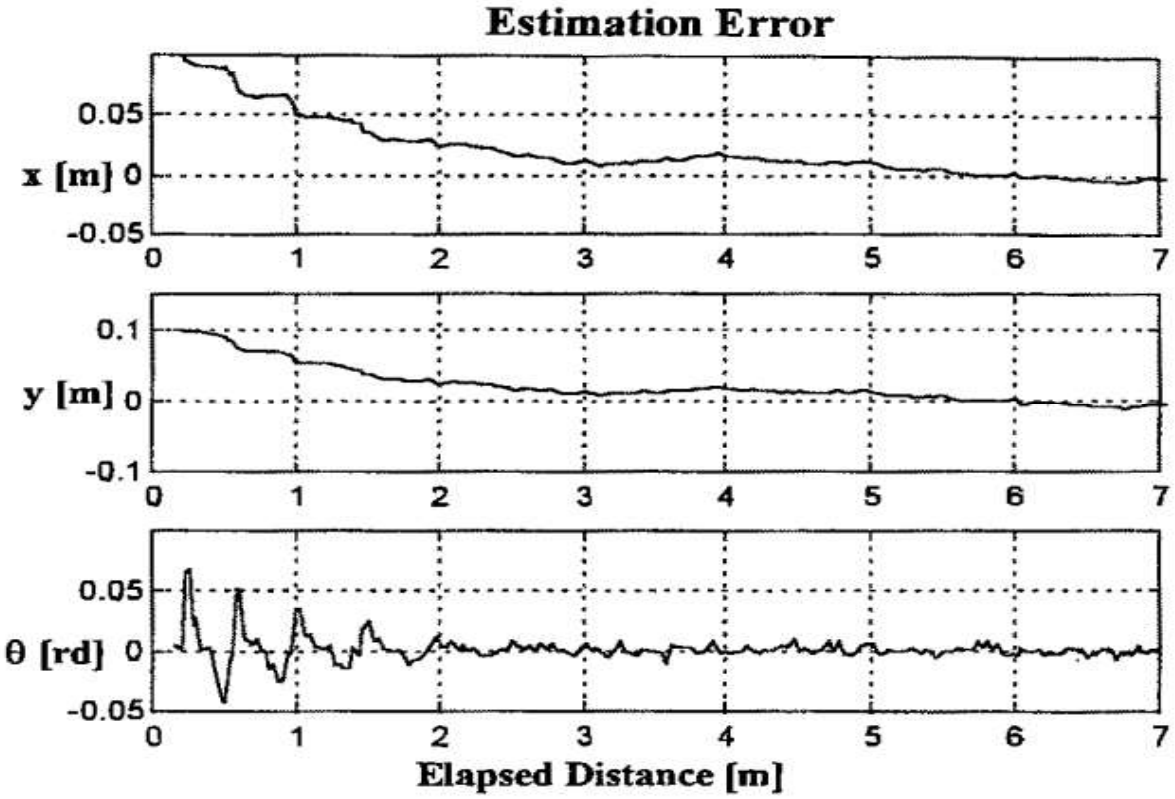

Fig9 Filter's Response To Initial Error In Position [1]

In the next and the final test, the Kalman filter is tested to the sensitivity of the beacon layout. It can be said that the effect of landmark layout is tested. During this sensitivity test, the vehicle was travelling on a straight trajectory. And along its trajectory several beacons were placed at different locations. The figure 10 shows the lateral error or deviation from the straight path. Following points were noted from the test:

- There are clear areas where the lateral error is larger than the normal value of error.
- On inspection it was found that the lateral errors increase when the beacons are in close contact with the mobile robot.
- Here close contact is meant by when two or more beacons and the robot are in one line.

- This can be explained by, if the lateral deviation is on the measurement line then the estimated angular measurement is same as the actual measurement.
- Because of this the filter is not able to detect the lateral error until several iterations, and it gets corrected after going through the iterations.

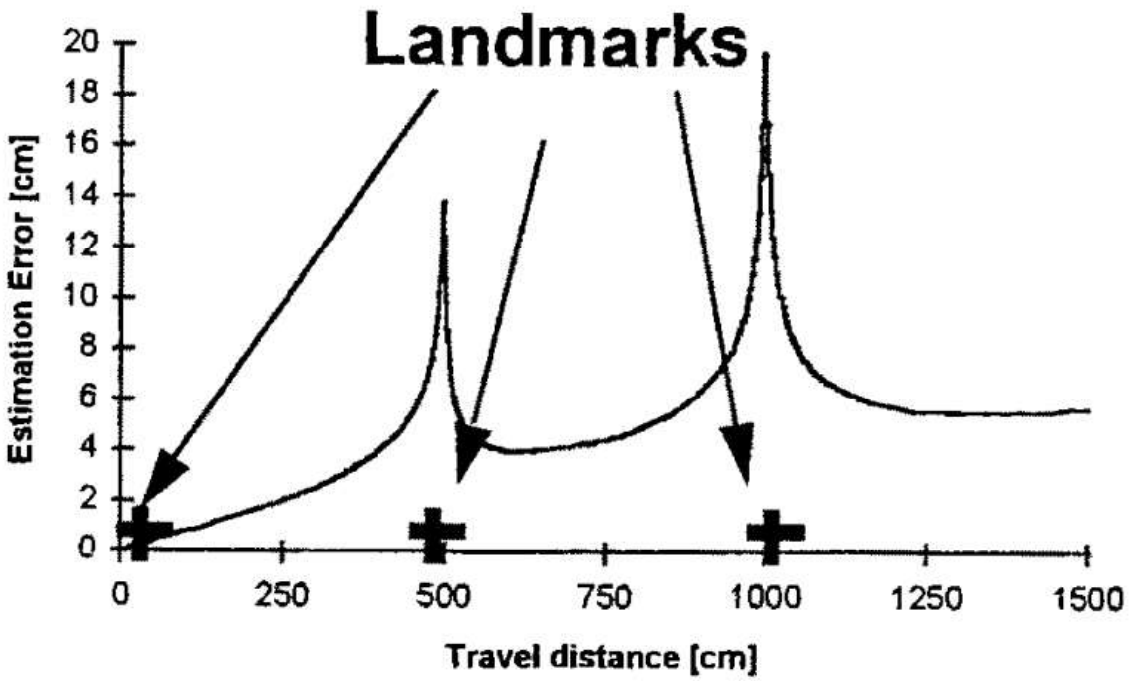

Fig10 Robot's deviation from straight path [1]

6 Implementation and Results

6.1 Python Code

This section implements the python code of the position estimation technique presented in the journal paper.

```
import numpy as np
from numpy.linalg import inv
import matplotlib.pyplot as plt


def getMeasurement(updataPara):
    if updataPara == 1:
        getMeasurement.currentPosition = 0
        getMeasurement.currentAngular = 60

    dt = 0.1
    w = 8 * np.random.randn(1)
```

```
    v = 8 * np.random.randn(1)
    z = getMeasurement.currentPosition + getMeasurement.currentAngular * dt +
v
    getMeasurement.currentPosition = z - v
    getMeasurement.currentAngular = 60 + w
    return [z, getMeasurement.currentPosition, getMeasurement.currentAngular]


def Kalmanfilter(z, updatePara):
    dt = 0.1
    # Initialize State
    if updatePara == 1:
        Kalmanfilter.x = np.array([[0],
                                   [20]])
        Kalmanfilter.P = np.array([[5, 0],
                                   [0, 5]])

        Kalmanfilter.A = np.array([[1, dt],
                                   [0, 1]])
        Kalmanfilter.H = np.array([[1, 0]])
        Kalmanfilter.HT = np.array([[1],
                                    [0]])
        Kalmanfilter.R = 10
        Kalmanfilter.Q = np.array([[1, 0],
                                   [0, 3]])

    # Predict State Forward
    x_p = Kalmanfilter.A.dot(Kalmanfilter.x)
    # Predict Covariance Forward
    P_p = Kalmanfilter.A.dot(Kalmanfilter.P).dot(Kalmanfilter.A.T) +
Kalmanfilter.Q
    # Compute Kalman Gain
    S = Kalmanfilter.H.dot(P_p).dot(Kalmanfilter.HT) + Kalmanfilter.R
    K = P_p.dot(Kalmanfilter.HT) * (1 / S)

    # Estimate State
    residual = z - Kalmanfilter.H.dot(x_p)
    Kalmanfilter.x = x_p + K * residual

    # Estimate Covariance
    Kalmanfilter.P = P_p - K.dot(Kalmanfilter.H).dot(P_p)

    return [Kalmanfilter.x[0], Kalmanfilter.x[1], Kalmanfilter.P]


def estimate():
    dt = 0.1
    t = np.linspace(0, 10, num=300)
    numOfMeasurements = len(t)
    num_landmarks = [1, 2, 3, 3.2, 3.1, 2.9, 3, 3, 3.2, 3, 2.8]
    error = [1.3, 2.1, 3.3, 4, 5, 6, 7, 8, 9, 13, 14]
    measTime = []
    measPos = []
    measDifPos = []
    estDifPos = []
    estPos = []
    estVel = []
```

```
    posBound3Sigma = []

    for k in range(1, numOfMeasurements):
        z = getMeasurement(k)
        # Call Filter and return new State
        f = Kalmanfilter(z[0], k)
        # Save off that state so that it could be plotted
        measTime.append(k)
        measPos.append(z[0])
        measDifPos.append(z[0] - z[1])
        estDifPos.append(f[0] - z[1])
        estPos.append(f[0])
        estVel.append(f[1])
        posVar = f[2]
        posBound3Sigma.append(3 * np.sqrt(posVar[0][0]))
    return [measTime, measPos, estPos, estVel, measDifPos, estDifPos,
posBound3Sigma, num_landmarks, error]


t = estimate()

plt.title('Sensitivity to Vehicle speed \n', fontweight="bold")
plot1 = plt.figure(1)
plt.xlim([0, 100])
plt.ylim([0, 500])
# plt.scatter(t[0], t[1])
plt.plot(t[0], t[2])
plt.ylabel('Speed')
plt.xlabel('Error')
plt.grid(True)



plot2 = plt.figure(2)
plt.plot(t[0], t[3])
plt.ylabel('theta')
plt.xlabel('Elapsed distance')
plt.title('Angular Position Estimation with Kalman Filter\n',
fontweight="bold")
plt.legend(['angular position estimation with filter'])
plt.grid(True)

plot3 = plt.figure(3)
# plt.scatter(t[0], t[4], color='grey')
plt.plot(t[0], t[5], color='grey')
plt.title('Linear position estimation with Kalman Filter \n',
fontweight="bold")
plt.plot(t[0], t[6])
plt.xlim([0, 100])
plt.legend(['Without filter', 'With Filter'])
plt.ylabel('Position points')
plt.xlabel('Elapsed Distance')
plt.grid(True)
plt.xlim([0, 300])

plot4 = plt.figure(4)
plt.title('Sensitivity to Number Of Landmarks \n', fontweight="bold")
```

```
plt.xlim([0, 15])
plt.ylim([0, 15])
plt.plot(t[8], t[7])
plt.ylabel('Error')
plt.xlabel('Number Of Landmarks')
plt.grid(True)
plt.show()
```

6.2 Results

This section discusses the results after implementing the position estimating technique.

In the first test the performance of kalman filter was tested against normal odometry estimation. The result is shown in figure11. It is evident that the performance of kalman filter with data fusion is very steady compared to the only odometry based estimation.

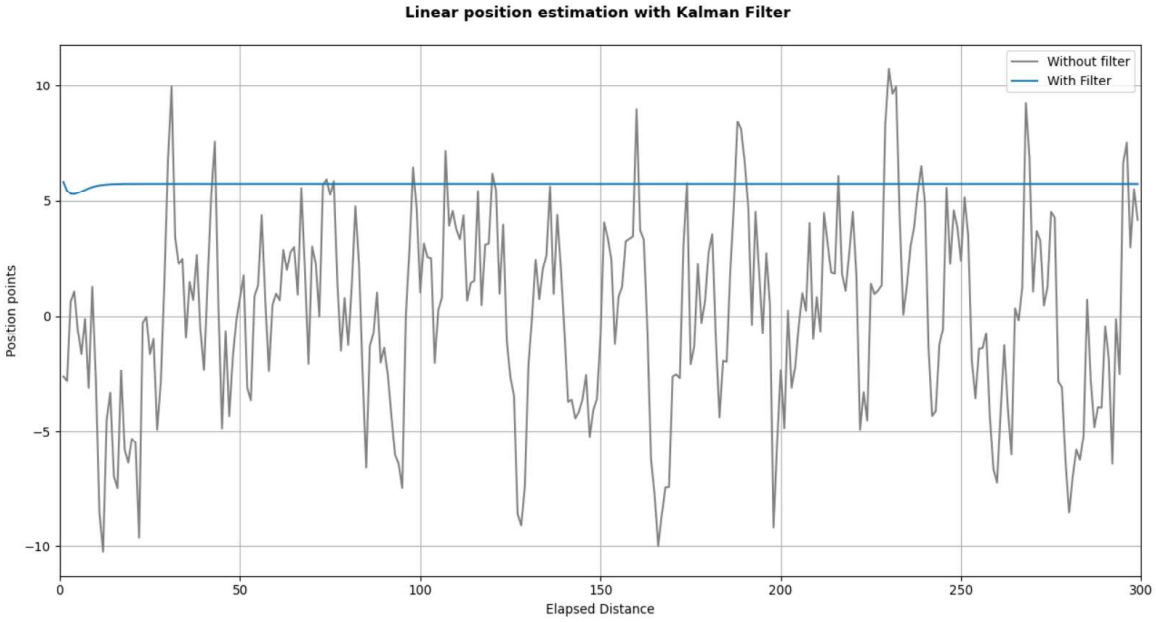

Fig 11 Kalman filter vs Normal Odometry

In the next test of the Kalman filter is tested against the speed i.e. filter's performance to the vehicle speed is analyzed. The following assumptions were made for the test:

- During the testing the mobile robot is allowed to traverse through a predetermined path.
- During each traversals the mobile robot is ran at different speeds.

The results are shown in figure 12. Following points were noted from the results:

- There is an evident relationship between the estimated position error and the vehicle speed.
- The relation between them is as expected, the error increases with the increase in the speed of the robot.
- This is because when the mobile robot travels at higher speeds, the robot doesn't perform angular measurement, which results in increased errors because of the increase in blind distance.

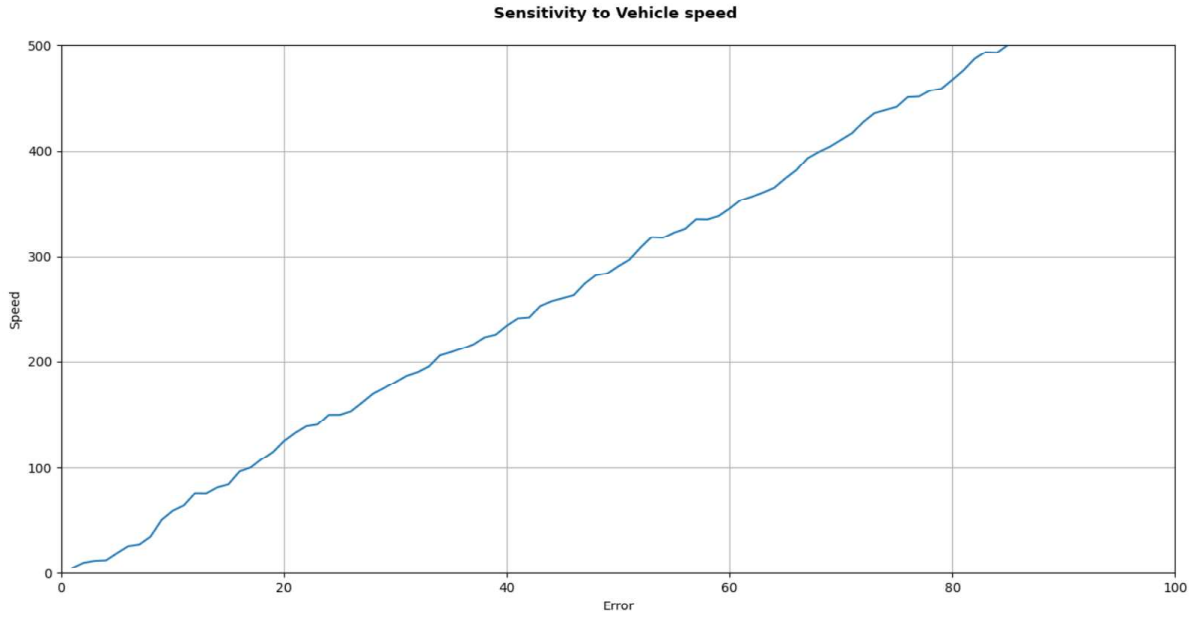

Fig12 Error vs Vehicle Speed

In the next test of the performance of Kalman filter is tested against the number of landmarks. Also, the number of landmarks used by filter is very crucial. Since, the mobile robot is expected to be travel in a completely unstructured environment, there are many scenarios where some of the landmarks would not be seen from each position i.e. some of the landmarks will be out of the sensor's detection range. Moreover, the cost involved in setting up the position estimating system can be reduced by reducing the number of landmarks, because the initial cost considers the set up cost, hardware cost and also precise estimation of the landmark.

It was assumed that during every travel the path was identical which helps to eliminate the effect of all other parameters except the landmarks. Figure 13 shows the results of this sensitivity test. The following points were noted from the test results:

- The effect is clearly more visible when the number of landmarks are less than 3.
- And the effect is more significant on the lateral position estimate and the effect on angular position estimate is minimal.

- Also, in case of more than three landmarks there is no significant effect on the angular as well as lateral position estimates.
- Moreover, the filter is able to give estimation on lateral errors only after having some number of iterations. And it requires 3 seconds to recover.
- While the filter can instantaneously detect the angular errors.

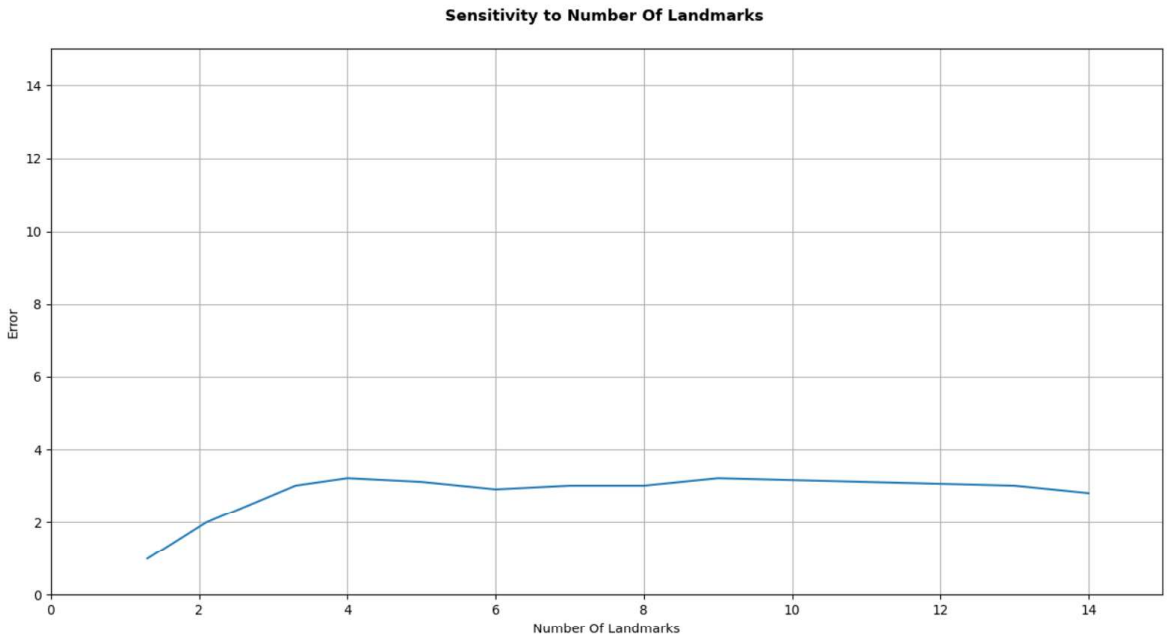

Fig13 Error vs Number of landmarks

6 Conclusion

With the advancements in the robotics and automation industry, more and more autonomous robots or autonomous vehicles have incorporated into the manufacturing industry, in the day to day life and also in the consumer industry. With these, it is expected that the autonomous vehicles doesn't pose any threat to the humans and their surrounding and also it is highly expected that the mobile robot perform the required task very accurately and precisely. And localization is one of the core components of the autonomous vehicle. Localization is the capability of an autonomous robot to precisely locate itself in the operating environment. For any autonomous vehicle it is very important to localize itself in order to carry out the required task. Many localization techniques have been evolved since last two three decades. In this report we discussed some techniques like odometry and triangulation for position estimation. But considering the accuracy of the required tasks and considering the practical uncertain working conditions, it was found that relying only on

these techniques is not enough. And a need of data fusion was certain. It was required to deduce a technique which can perform estimation very well by fusing the data from the techniques like odometry and triangulation. And it was found that Kalman filter acts as one of the best estimating techniques. And the data coming from odometry and triangulation was fused into the Kalman filter to carry out accurate position estimation.

On testing the algorithm based on the real implementation and from the results presented in the journal paper, one of the significant conclusions was that the filter performs better when the position error is in terms of orientation compared to the lateral position error. Also, the filter was able to offset the error within 3 seconds. Moreover, it can be deduced that error increases linearly with increase in speed. And also the number of beacons and their orientation can significantly affect the performance of the filter. Hence, it can be concluded that the kalman filter based technique presented in the selected journal paper undoubtedly has a robust performance, but it also has some limitations with respect to the speed of the mobile robot as well as the number of beacons and their layout. Finally, there are also some advancements in the kalman filter like extended kalman filter and unscented kalman filter which may significantly improve the performance as well.

7 References

[1] S. Shoval, I. Zeitoun, and E. Lenz, “Implementation of a Kalman filter in positioning for autonomous vehicles, and its sensitivity to the process parameters,” *The International Journal of Advanced Manufacturing Technology*, vol. 13, no. 10, pp. 738–746, Oct. 1997.

[2] S. Shoval, I. Zaietun and E. Lenz. "Layout of beacons for triangulation of AGV's in industrial environment", Proceedings of the 13th International Conference on Production Research, Jerusalem, Israel, pp. 485-488, August 1995.

[3] R. E. Kalman, “A New Approach to Linear Filtering and Prediction Problems,” J. Basic Eng., vol. 82, no. 1, p. 35, 1960.

[4] M. Tang, Z. Chen, and F. Yin, “Robot Tracking in SLAM with Masreliez-Martin Unscented Kalman Filter,” *International Journal of Control, Automation and Systems*, vol. 18, no. 9, pp. 2315–2325, May 2020.

[5] Q. Li, R. Li, K. Ji, and W. Dai, “Kalman Filter and Its Application,” in 2015 8th International Conference on Intelligent Networks and Intelligent Systems (ICINIS), 2015, no. 10, pp. 74–77.

[6] Hidekata Hontani and Yuya Higuchi, “Kalman filtering for improving car positioning with measurements of relative distance,” *SICE Annual Conference 2007*, Sep. 2007, doi: 10.1109/sice.2007.4421490.

[7] M. Wang, H. Hu, X. Wang, H. Zhang, and A. Zhang, “An improved method of adaptive Kalman filtering for vehicular kinematic positioning,” *Proceedings of the 33rd Chinese Control Conference*, Jul. 2014, doi: 10.1109/chicc.2014.6896726.

[8] U. Wiklund, U. Andersson and K. Hyyppa, "AGV navigation by angle measurements", Proceedings of the 6th International Conference on Automated Guided Vehicle Systems, pp. 199-212, October 1988.

[9] “Understanding the Kalman Filter,” *The American Statistician*, 2012. https://www.tandfonline.com/doi/abs/10.1080/00031305.1983.10482723.

[10] H.-J. von der Hardt, P. Arnould, D. Wolf and M. Dufant, "A method of mobile robot localization by fusion of odometrical and magneto metric data", International Journal of Advanced Manufacture Technology, pp. 65--69, 1994.

[11] H. Chen, H. Lim, S. Hwang, and J. Lee, “Efficient driving of tracking robot using kalman filters,” *2013 13th International Conference on Control, Automation and Systems (ICCAS 2013)*, Oct. 2013, doi: 10.1109/iccas.2013.6703874.

[12] Wikipedia Contributors, “Automated guided vehicle,” *Wikipedia*, Nov. 02, 2021. https://en.wikipedia.org/wiki/Automated_guided_vehicle.