3.1. Introduction

Sensors used in SLAM algorithms measure both the internal parameters of the robot and the external environmental features. These measured parameters will have inferences about the robot location and map of the environment, which are required for SLAM implementation. Hence, there is a need for signal processing algorithms, which can extract the localization and mapping information from the sensor measurements and also account for their measurement errors.

Odometer sensor measures the wheel displacement. It can be used to locate the robot position by accumulating the wheel displacement with its initial position. This process is popularly known as ‘dead reckoning’ [7, 36]. It is one of the most preferred techniques for WMR localization because of its simplicity and ease of use. Being a relative measurement technique, even a small measurement error will deteriorate the localization accuracy of dead reckoning. Hence, there is a need for signal processing techniques to eliminate the measurement error.

Wheel slip [36-38] is one of the significant perturbations that cause measurement error in dead reckoning. It can be defined as the mismatch between the wheel displacement and the actual movement of the robot. The wheel slip depends on various factors such as wheel-terrain interaction, wheel velocity, acceleration, dimension and material of the robot wheels [65-67]. Other than wheel-terrain interaction, rest of the factors are either measurable or remains constant, which can easily be compensated.
The wheel-terrain interaction mainly depends on the nature of terrain and robot wheels [67]. A hard surfaced terrain will increase the wheel displacement than the actual robot movement due to increased friction, whereas on a soft, slippery surface the actual robot movement will be higher than the wheel displacement as the robot tends to slide due to lesser friction. Thus, the odometer sensor is in need of terrain specific signal processing algorithms to compensate these wheel slip, which can improve the accuracy of dead reckoning.

This chapter deals with the,

- A mathematical model of the odometer, which can translate the odometer measurement into robot location
- Development of slip compensation techniques using the neuro-fuzzy system and stochastic approach
- Design of terrain classification system using the adaptive neuro-fuzzy system to produce terrain specific slip compensation
- Finally, the experimental results and their inferences are discussed

### 3.2. Odometer Model

The odometer sensor is a high-speed optical shaft encoder as shown in Figure 3.1, which measures the wheel displacement in the form of a quantized distance called ‘ticks’ ($d_{tic} \in \mathbb{Z}$). The velocity of the robot wheels can be calculated using rate of change of these tick values and a tick conversion factor ($T_{cf}$), which converts the ticks into the corresponding displacement. The tick conversion factor is calculated as a ratio of wheel displacement to the number of ticks ($R_{tic}$) per revolution as in Eq. (3.1). The WMR used
in the proposed work has a wheel radius \( (r_w) \) of 0.0552 m, and it displaces 3200 ticks per revolution.

The velocity of the left wheel \( (V_{lw}) \) and right wheel \( (V_{rw}) \) of the WMR are calculated using the measured left wheel \( (ld_{tie}) \) and right wheel ticks \( (rd_{tie}) \) respectively as in Eq. (3.2). Using the measured wheel velocities, the change in robot pose \( (U^n_r) \) in terms of the relative translation \( (\Delta T_r) \) and rotation/orientation \( (\Delta \varphi_r) \) is determined with the cross-section length of WMR \( (R_t = 0.32 \text{ m}) \) as in Eq. (3.3). Thus, the slip compensation system has to produce a compensation velocity, which can compensate the effect of wheel slip in the measured left and right wheel velocity.

\[
T_{cf} = \frac{2\pi r_w}{R_{tie}} = 0.01086 \tag{3.1}
\]

\[
\begin{bmatrix} V_{lw} \\ V_{rw} \end{bmatrix} = \frac{T_{cf}}{\Delta t} \begin{bmatrix} \Delta ld_{tie} \\ \Delta rd_{tie} \end{bmatrix} \tag{3.2}
\]

\[
U^n_r = \begin{bmatrix} \Delta T_r \\ \Delta \varphi_r \end{bmatrix} = \begin{bmatrix} 1/2 & 1/2 \\ R_t/2 & -R_t/2 \end{bmatrix} \begin{bmatrix} V_{lw} \\ V_{rw} \end{bmatrix} \tag{3.3}
\]
3.3. Slip Compensation System

The odometer sensor requires efficient slip compensation technique, which can improve the accuracy of dead reckoning. The variable nature of wheel-slip from one robot to another, and its dependency on various factors such as wheel-terrain interaction, wheel velocity, acceleration, etc., demand intelligent techniques to be used for developing slip compensation systems.

In literature, several approaches have been reported for slip compensation [8]. A multi-model based framework has been proposed [37-38], wherein, several models are built based on the slip conditions using conventional or intelligent techniques. These models are used to predict the nature of wheel slip and also to take corrective action accordingly. The models are even switched based on the terrain nature to account for the terrain dependent nature of wheel slip. Hence, there is a need for automated/self-learning techniques to create these slip compensation models. Further, the complex nature of wheel-terrain interactions demands stochastic slip compensation models [66] to account for the unstructured measurement noises. Thus, in the proposed work, two different methodologies have been investigated to develop slip compensation system as follows,

a) Neuro-Fuzzy System (NFS) based Self Calibration Technique

b) Stochastic Switching Model-based Technique

3.3.1. Neuro-Fuzzy System (NFS) based Self Calibration Technique

The complex nature of wheel slip and their dependencies on various factors poses significant challenges before the research community. This problem can be addressed using intelligent learning techniques for developing a slip compensation system. Once the

---

slip has been compensated, the robot can be accurately localized using dead reckoning sensor with the tuned slip compensation model alone, thereby eliminating the need for additional filters and additional sensors as in other localization techniques. Though this approach offers significant advantages, its main drawback is that the tuned slip compensation model may fail to capture the environmental variations. One way to make the method robust is to recalibrate [68-69] the compensation models considering the changes in the environment.

Motivated by this research gap, this investigation aims at developing a wheel-slip compensation technique that uses self-calibration approach. The proposed self-calibration technique uses NFS for calibration of the odometer and for building accurate wheel slip compensation model. It has been pointed out in [66] that the wheel slip depends on environmental parameters such as terrain characteristics and wheel-terrain interaction. Additionally, the robot parameters such as wheel pressure, the center of mass, robot velocity, type of motion, etc. will also influence the slip compensation technique. These parametric dependencies make the wheel slip compensation design to be challenging using analytical methods.

On the other hand, data collected from experiments containing intrinsic information about these parameters can also be exploited to build compensation models. Furthermore, the compensation models need to be adapted for environmental conditions. NFS that combine the learning features of neural networks with decision-making capabilities of fuzzy systems in uncertainties emerges as a good candidate for this problem.
3.3.1.1. Proposed Self Calibration Technique

Self-calibration [69] is an automated procedure in which the signal conditioning unit of the sensor system is configured to minimize the error between the actual physical quantity and its measurement as in Figure 3.2. To perform calibration, the sensor system is subjected to measure the prior known or accurately measured physical quantity called standard data ($X_s$). The measurements made by the sensor system are called as measured data ($X_m$). The error between these data is used by the calibration system to configure the conditioning unit. The configured signal conditioning unit will compensate for the error to reduce the deviation between the standard and measured data.

![Diagram of Self-calibration system](image)

Figure 3.2. Self-calibration system.

In the proposed work, odometer sensor and slip compensation model is considered as sensing and conditioning unit respectively as shown in Figure 3.3. Laser range finder (LRF) based localization [70] with the known landmarks can precisely
estimate the robot location. Hence, it is used to generate standard data in 4 steps as follows,

In the first step, the LRF scans the environment to acquire the relative position of the landmarks with reference to the robot \( X_l \). The coordinates of the landmarks are extracted in feature extraction step. In data association step, the extracted landmarks across each scan are associated to estimate the scene change. Using the estimated scene change, the robot location can be calculated using a mathematical model of the WMR in the final step.

The odometer and its mathematical model (Ref: Section 3.2) are used to generate the measured data, which are subjected to error created by wheel slip. The localization error due to wheel slip is calculated by using the deviation of measured data from the standard one. This error can be compensated by designing a fuzzy based slip compensation model. The learning capability of neural networks in NFS acts as a calibration system, which learns the dynamics of the slip-induced localization error. This knowledge learned by the neural networks is imposed into a fuzzy logic system to create a slip compensation model, which can compensate for the slip error in odometer measurement.

3.3.1.2. Generation of Standard Data

The standard data for localization are generated using LRF with the known landmarks, which are kept in the known location at robot’s vicinity. To localize the WMR precisely, the experiments are conducted on a rectangular arena with four cylindrical pillars placed at each corner that act as landmarks. To enable the robot to distinguish landmarks, they are set to have variable cross-section area as in Figure 3.4.
The robot is set on a random motion in the arena ensuring at least one landmark will be in robot’s vicinity throughout the run-time. It enables the robot to be localized accurately concerning the visible landmark and also to generate standard data.

Figure 3.3. Proposed self-calibration of odometer sensor.

Figure 3.4. Test arena for self-calibration of odometer sensor.
The relative position of the landmarks with respect to the robot’s center can be extracted from LRF scans. In LRF scans, the landmarks appear as point clusters indicating a part of cross section of landmarks. It subjects the scanned surface to fall into these two categories as follows,

- **Category I:** when the relative position of robot and landmark is large, the LRF will have better resolution, and the landmark cross-section appears like an arc (Ref: Case:1 in Figure 3.5)

- **Category II:** when the robot is near the landmark, the resolution of LRF degrades due to lesser time-of-flight, which causes the landmarks to appear as a straight line (Ref: Case:2 in Figure 3.5)

However, in both categories, the center of the landmark is estimated by orthogonally projecting the robot barreling along the center of the scanned data as illustrated in Figure 3.5. Thus, the use of these structured landmarks makes it easier to extract its coordinates accurately in any circumstance.

The extracted landmark coordinates \((r_a, \theta_a)\) between the consecutive LRF scans are associated to determine scene change in terms of translation \((\Delta T_x)\) and rotation \((\Delta \phi_x)\) as in Eq. (3.4) and Eq. (3.5) respectively. The landmarks are associated \((N_a - \text{the number of associated landmarks})\) by using nearest neighborhood search and Quickhull algorithm [71] without tessellation. This ensures only the common landmarks between the scans are associated. It is advantageous to have faster LRF scan rate as compared to the robot navigation speed, which ensures the existence of at least one common landmark between two successive LRF scans.
\[ \Delta T_r = -\frac{1}{N_a} \sum_{a=1}^{N_a} (r_a^n - r_a^{n-1}) \] (3.4)

\[ \Delta \varphi_r = -\frac{1}{N_a} \sum_{a=1}^{N_a} (\theta_a^n - \theta_a^{n-1}) \] (3.5)

**Figure 3.5. Estimation of landmark’s center.**

### 3.3.1.3. Slip Compensation Model Using NFS

The deviation between the standard data (using LRF) and the measurement data (using odometer) is used to calculate the wheel slip. With this wheel slip as training data, NFS is used to learn the slip dynamics and also to create a slip compensation model. NFS is a hybrid technique, which combines the learning capability of the artificial neural network (ANN) with the knowledge representation capability of fuzzy inference system (FIS) [35, 37, 72]. It makes NFS an ideal tool to learn the intricate slip dynamics from the wheel slip data and also to represent the knowledge for slip compensation. Table 3.1
illustrates the details of various parameters used in NFS to design the slip compensation model.

Thus, the resultant FIS based slip compensation model is used to compensate the slip occurred during robot navigation and thereby improved the accuracy of the odometer. As the WMR navigates in a two-dimensional plane, two slip compensation models ($FIS_x, FIS_y$) have been designed to compensate localization error due to wheel slip on each dimension. The designed FIS uses parameters [73] such as velocity along each dimension($V_x, V_y$), its acceleration component ($A_x, A_y$) and, the absolute difference in wheels velocities ($|V_{lw} - V_{rw}|$) as inputs to determine the compensation velocity($\Delta V_{cx}, \Delta V_{cy}$) as in Figure 3.6. These compensation velocities are subtracted from the measured robot velocity to eliminate wheel slip, and the robot location is estimated as in Eq. (3.6).

<table>
<thead>
<tr>
<th>Table 3.1. Design specifications of the neuro-fuzzy system.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Parameters</strong></td>
</tr>
<tr>
<td>Type of Fuzzy Inference System</td>
</tr>
<tr>
<td>No of Inputs</td>
</tr>
<tr>
<td>Types of Input Membership Function</td>
</tr>
<tr>
<td>Type of Output Function</td>
</tr>
<tr>
<td>Defuzzification Method</td>
</tr>
<tr>
<td>Learning Method</td>
</tr>
</tbody>
</table>

\[
X^n_r = \begin{bmatrix} x^n_{r-1} \\ y^n_{r-1} \\ \varphi^n_{r-1} \end{bmatrix} + \begin{bmatrix} \Delta T_r \cos \varphi^n_{r-1} \\ \Delta T_r \sin \varphi^n_{r-1} \\ \Delta \varphi_r \end{bmatrix} - \begin{bmatrix} \Delta V_{cx} \\ \Delta V_{cy} \\ \tan^{-1}(\Delta V_{cy}/\Delta V_{cx}) \end{bmatrix}
\] (3.6)
3.3.1.4. Results and Discussion

The performance of the proposed slip compensation model is validated by using two sets of data namely (i) training and (ii) testing data, which are generated from different experiments. Experimental data are generated by setting the robot to perform closed loop navigation with different linear and rotation velocities.

The training data are used by NFS to design the slip compensation model for the specified environment. The performance validated using this training data will evaluate the learning ability of the NFS to learn the slip dynamics. Testing data are generated with a different set of experiments, and it validates the adaptive capability of NFS to compensate slip under different conditions.

Figure 3.6. FIS for slip compensation.

Figure 3.7 illustrates the performance of the NFS based self-calibrated odometer for training and testing data. It is observed that an uncalibrated sensor encounters the slip error, which gets accumulated as the robot navigates whereas the calibrated sensor compensates for wheel slip in terms of velocities as in Figure 3.7(a & c). So that the slip
error is compensated at every instant of time and thereby the drift in localization has been minimized as seen in Figure 3.6(b & d).

Figure 3.7. Performance evaluation for training data (a), (b) & (e) and for testing data (c),(d)&(f). (blue - indicates the uncalibrated odometer, green - indicates the standard data from LRF and red - indicates calibrated odometer using NFS).
Further, the performance of the proposed system is evaluated in terms of localization error, which is a measure of the deviation between estimated and actual robot location as in Figure 3.6(e & f). It has been observed that the maximum localization error occurred for both data is around 0.28 m, which is much lesser than the robot length \(R_l = 0.47\, \text{m}\). This ensures that the calibrated odometer is able to keep track of the robot position at every point of time without any need for additional sensors. A mean square localization error (MSE) of 0.0382 m for a 6.12 m run and 0.0442 m for a 4.20 m run during training and testing phase respectively has been observed, which illustrates the performance of the proposed self-calibration technique.

3.3.2. Stochastic Switching Model-based Slip Compensation Technique

The uncertainties associated with wheel slip and their dependences on various factors make slip compensation technique a challenging task. On the other hand, experimental data provides valuable information to model slip [66]. Therefore, models that use measurement data to capture uncertainty in wheel slip compensation technique are required. Hence, the proposed work uses a combination of linear and Gaussian distribution to model wheel slip under various conditions such as variations in robot motion and terrain type in which the robot navigates. Linear models can be used to compensate for the slip dynamics that depends on the measurable factors like wheel velocity, acceleration, etc., whereas Gaussian models are used to capture the uncertainties present in the wheel slip.

These models are used to compensate the slip that is generated in the odometer while measuring the robot wheel velocity \(V_{wm}\). They are used to create a compensation

velocity \( (V_{sc}) \), which will be subtracted from the measured wheel velocity. Finally, the robot pose is estimated using the odometer model illustrated in section 3.2. The slip compensation models are selected according to the type of robot motion (Ref: Table 3.2). The type of motion is detected using the input command (Duty ratio given to the left \( (D_{lw}) \) and right \( (D_{rw}) \) wheel of the robot) and the type of terrain is identified by using both the input command and output wheel velocity as illustrated in Figure 3.8.

**3.3.2.1. Experimental Procedure to Acquire Wheel Slip**

Wheel slip causes perturbations in robot velocity measurements leading to performance degradation of dead reckoning based localization methods. The disturbances in measurements can be detected by knowing the actual velocity of the robot accurately, which in turn requires reliable measurement technique. To obtain such measurement, this investigation uses a binary type of measurement technique using distance measurement unit (DMS) with a simple form of feature extraction and data association methods. Being a binary type measurement, the proposed technique is meant to be accurate and reliable.

![Figure 3.8. Stochastic switching model-based technique for slip compensation in odometers.](image-url)
DMS is an infrared (IR) based reflection type sensor, which is fitted focusing outwards along the vertical axis of each wheel. The robot is set to be in any one of the motion as described in Table 3.2 and along its path, a reflective surfaced reference point is placed such that that the DMS senses it as an obstacle per rotation as shown in Figure 3.9. The number of rotation measured by the DMS over the given time is used to determine the actual velocity of the robot. The deviation between the actual robot velocity and the odometer measured velocity is calculated to determine the wheel slip.

Table 3.2. Types of robot motion.

<table>
<thead>
<tr>
<th>Motion Type</th>
<th>Control Signal</th>
<th>Center of Rotation</th>
<th>Direction of Rotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left wheel forward (LF)</td>
<td>$D_{lw} &gt; 0$</td>
<td>Right Wheel</td>
<td>Clockwise</td>
</tr>
<tr>
<td>Left wheel reverse (LR)</td>
<td>$D_{lw} &lt; 0$</td>
<td>Right Wheel</td>
<td>Anti-clockwise</td>
</tr>
<tr>
<td>Right wheel forward (RF)</td>
<td>$D_{lw} = 0$</td>
<td>Left Wheel</td>
<td>Clockwise</td>
</tr>
<tr>
<td>Right wheel reverse (RR)</td>
<td>$D_{lw} = 0$</td>
<td>Left Wheel</td>
<td>Anti-clockwise</td>
</tr>
<tr>
<td>Robot Forward</td>
<td>$D_{lw} &gt; 0$</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>Robot Reverse</td>
<td>$D_{lw} &lt; 0$</td>
<td>---</td>
<td>---</td>
</tr>
</tbody>
</table>

Figure 3.9. Robot motion for estimating the wheel slip.
3.3.2.2. Linear Gaussian models

A non-recursive linear Gaussian model is considered to be an optimal model to capture the wheel slip dynamics and its uncertainties. As the wheel slip depends on the wheel velocity, the model has a biased linear relation, and unbiased Gaussian noise is used to account for their uncertainties as in Eq. (3.7).

\[ V_{sc}^n = A_0 V_{wm}^n + B_0 + \omega_s^n \] (3.7)

In Eq. (3.7), \( \omega_s^n \approx N(0, S) \) is the Gaussian distribution modeling the slip with zero mean and covariance (S). The vectors \( A_0 \) and \( B_0 \) denote the coefficient and bias respectively. The linear trend in the acquired wheel slip data with respect to the measured wheel velocity is fitted with a first-order polynomial using least square estimate [74]. The linear models thus generated provide the remaining wheel slip noise (\( \omega_{gs} \)) that can be modeled using a Gaussian distribution as described by the Algorithm 3.1.

---

**Algorithm 3.1:** Wheel slip modeling algorithm

**Data:** \( V_{sc}, V_{wm} \)

**Return:** \( A_0, B_0, S \)

**Initialize:** \( MT = \{LF, LR, RF, RR\} \)

**For each** \( MT \) **do**

- Compute: \([A_0, B_0] \leftarrow \text{LinearFit}(V_{sc}, V_{wm})\)
- Return: \([A_0, B_0]\)
- Compute: \( \omega_s = V_{sc} - (A_0 V_{wm} + B_0) \)
- Compute: \( [S] \leftarrow \text{GaussianFit}(\omega_s)\)
- Return: \([S]\)

**End for**
For each type of motion (MT), a stochastic slip compensation model is generated and the models will be switched according to the input command. Figure 3.10 illustrates the wheel slip data and its corresponding model for various types of robot motion. It is observed that the models are able to fit in the slip data with minimal error.

![Figure 3.10. Linear Gaussian models for slip compensation.](image)

**3.3.2.3. Results and Discussion**

To investigate the performance of the proposed slip compensation models, this study compares the experimental results of the uncompensated and the compensated mapping technique. As mapping and localization are interrelated problems, the quality of the reconstructed map can act as a reliable indicator to evaluate the localization accuracy.
Thus, in the proposed work the quality of the reconstructed map is evaluated, which validates the accuracy of slip compensation model in localizing the WMR.

In the second test, the proposed multi-model based switching technique is validated based on its ability to perform localization and mapping by subjecting the robot into a random rotation motion as shown in Figure 3.12. The desired robot track has been determined based on the time at which the robot switches its type of motion, and the results demonstrate the ability of the proposed slip compensation in reducing the tracking error as in Figure 3.12c.

Table 3.3. Real-time test conditions for slip compensation models.

<table>
<thead>
<tr>
<th>Test Conditions</th>
<th>Slip Compensation Model</th>
<th>Type of Robot Motion</th>
<th>Observation</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>Individual Model</td>
<td>Rotational Motion</td>
<td>Mapping performance of four models have been tested individually as in Figure 3.10</td>
</tr>
<tr>
<td>II</td>
<td>Multi-model Switching</td>
<td>Rotational Motion</td>
<td>Mapping and localization performance of the models have been evaluated as in Figure 3.11</td>
</tr>
<tr>
<td>III</td>
<td>Multi-model Switching</td>
<td>Linear and Rotational Motion</td>
<td>Mapping and localization performance of the models have been evaluated for all possible motion of WMR as in Figure 3.12</td>
</tr>
</tbody>
</table>

The third test condition studies the performance of the proposed slip compensation technique when the robot is in linear motion. While in forwarding motion, the proposed technique uses LF and RF slip compensation model to compensate the slip in left and right wheel of the robot respectively. Whereas in reverse motion, LR and RR models are used for slip compensation. The performances of these slip compensation models are evaluated by subjecting the robot to perform a loop closure in a narrow arena,
where linear motion is predominant. From Figure 3.13, it is observed that the proposed slip compensation technique can reduce the deviation between the estimated and the actual robot track, thereby, improving the localization accuracy.

Figure 3.11. Mapping performance of the slip compensation models.

Further, the mapping performance of the proposed slip compensation technique is quantified using a proposed metric called ‘Map spread factor’ (MSF). MSF is a measure of relative occupancy of the obstacles to the free space in the constructed map. The idea behind MSF formulation is that an inaccurate localization will lead to a spread of
estimated obstacle locations and it causes an increase in MSF, which indicates a more significant error in the constructed map. On the other hand, an accurate localization reduces spread of obstacle position and MSF, which will report an accurate mapping.

MSF can be calculated by converting the map into a pixilated binary image with the specified resolution \( M_r \) into black and white pixels. A white pixel corresponds to an obstacle, and a black indicates a free space. The acquired scan data \( X_m \) from the LRF will be level shifted to quantize into a digital binary image. The cardinality of the white pixel and black pixel are calculated, whose ratio determines the MSF as described in the Figure 3.14. In other words, MSF gives the relative degree of obstacle occupancy and the free space in the constructed map.

![Figure 3.12](image)

(a) Uncompensated map   (b) Compensated map   (c) Robot track

Figure 3.12. Localization and mapping performance of proposed technique during rotation motion.
Even though MSF cannot directly measure the mapping efficiency, but it can provide a measure to study it. To obtain a measure for improvement in map reconstruction, we define the new measure called ‘percentage mapping improvement’ \((M_{imp})\), defined as in Eq. (3.8). It is defined as the ratio of a difference between the MSF’s of uncompensated \((MSF_u)\) and compensated map \((MSF_c)\) to the MSF of an uncompensated reconstructed map.

Table 3.4 illustrates the performance of the proposed slip compensation and its improvement in terms of its mapping ability. It demonstrates that the proposed slip compensation scheme improved the mapping efficiency by around 72% during rotation and by 67 % during linear motion over the uncompensated odometer. The proposed slip
compensation system not only improves the accuracy of dead reckoning, but the realization is also cheap and straightforward. Extension to adaptive slip compensation technique and testing the proposed methods in various deployment environments are the future course of this investigation.

\[ M_{im} = \left\{ [x_n, y_n] \mid x_n \in [x_{min}, x_{max}], \ y_n \in [y_{min}, y_{max}] \right\} \]

\[
\begin{bmatrix}
    x_{sm} \\
    y_{sm}
\end{bmatrix} = \begin{bmatrix}
    x_m \\
    y_m
\end{bmatrix} - \begin{bmatrix}
    x_{min} \\
    y_{min}
\end{bmatrix} + M_i
\]

\[
M_B(r, c) = M_B\left(\frac{x}{M_r}, \frac{y}{M_r}\right) = \begin{cases}
1 & x \in x_{sm}, y \in y_{sm} \\
0 & \text{else}
\end{cases}
\]

\[
W_p = \left| \{ M_B(r, c) \mid M_B(r, c) = 1 \} \right|
\]

\[
B_p = \left| \{ M_B(r, c) \mid M_B(r, c) = 0 \} \right|
\]

\[
MSF = \frac{W_p}{B_p}
\]

Figure 3.14. Flow-chart to calculate MSF.

\[
M_{imp} = \frac{MSF_u - MSF_c}{MSF_u} \times 100\% \quad (3.8)
\]
Table 3.4. Performance validation of stochastic slip compensation technique.

<table>
<thead>
<tr>
<th>Test Condition</th>
<th>Type of Motion</th>
<th>$MSF_u$</th>
<th>$MSF_c$</th>
<th>$M_{imp}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>Left wheel forward (LF)</td>
<td>1.5529</td>
<td>0.3994</td>
<td>74.28</td>
</tr>
<tr>
<td></td>
<td>Left wheel reverse (LR)</td>
<td>0.5776</td>
<td>0.4073</td>
<td>29.48</td>
</tr>
<tr>
<td></td>
<td>Right wheel forward (RF)</td>
<td>1.0753</td>
<td>0.3951</td>
<td>63.26</td>
</tr>
<tr>
<td></td>
<td>Right wheel reverse (RR)</td>
<td>1.2107</td>
<td>0.2591</td>
<td>78.60</td>
</tr>
<tr>
<td>II</td>
<td>Rotation Motion</td>
<td>1.3410</td>
<td>0.3681</td>
<td>72.55</td>
</tr>
<tr>
<td>III</td>
<td>Linear and Rotation Motion</td>
<td>0.6747</td>
<td>0.2246</td>
<td>66.71</td>
</tr>
</tbody>
</table>

3.4. ANFIS based Terrain Classifier

Wheel slip depends on the nature of terrain [66-67] in which the robot navigates, and it becomes vital to know terrain characteristics for developing slip compensation mechanisms. Hence, the proposed work aims to build intelligence in WMR to classify terrains [75] using simple features and onboard sensors available in the robot. The impact of wheel-terrain interaction [66] on the wheel behavior has been used as features to classify terrain. It can improve the robot navigation by providing terrain specific slip compensation, and have customized control over the robot wheels.

Seeing the importance of the terrain classification problem, researchers have investigated the problem, and many approaches have been proposed. The available approaches can be broadly classified as vision (non-contact type), and reaction based (contact type) [75]. In vision-based terrain classification methods, vision sensors such as motion, aerial, and monocular cameras [76] are used for classifying terrains in a large scale environment, which demands complex and powerful computations. The reaction based classification technique relies on additional sensors such as inertial measurement

---

unit (IMU), pressure arrays, and contact sensors [77] for terrain classification. However, it is limited to the terrain surface in which the sensor comes in contact.

The existing techniques encounter two major problems namely, (i) complex computations required for performing feature extraction and (ii) sensor imprecision and disturbance. Furthermore, terrain classification requires learning from raw sensor data and making decisions on the correct type of terrain. It demands techniques that are capable of learning from the observed measurements and also to make decisions with such imprecise data.

Traditionally, ANN has emerged as a promising tool for learning from data, whereas they lack decision making capability. On the other hand, FIS can make decisions in the presence of imprecise or data corrupted by noise. However, they lack learning capabilities. Therefore, terrain identification requires a fusion of learning and decision-making capability. This investigation uses adaptive Neuro-fuzzy inference system (ANFIS) that provides a hybridization of ANN and FIS. As a result, ANFIS can learn from historical sensor information and make decisions based on the assimilated knowledge. These are essential properties to build perception mechanism in WMR for classifying terrains.

3.4.1. Terrain Classification Problem Description

The problem studied in this investigation is the development of a perception mechanism to classify terrain type in WMR. In particular, the role of combining experimental raw data with data mining methods to create knowledge and embedding such assimilated knowledge to build a perception mechanism that can classify terrains on-the-move is explored.
Three factors motivate the selection of ANFIS as a tool to build perception mechanism. First, the accuracy of the terrain classification algorithms depends on the wheel-terrain interactions that varies with the type of the robot and the hardware used. Even within the same robot type, customizing becomes vital due to the heterogeneity of hardware, software, and the external conditions. Consequently, terrain classification algorithms [75] need to be tailored to a particular robot. Alternatively, perception mechanism using raw data for building intelligence can be adapted easily for a specific type of robot as they process patterns of data.

Second, the prior knowledge from experiments/simulations can be embedded in building the perception mechanism. It demands the technology, which can exploit the power of learning capability and also can develop intelligence. Third, learning from data can be used to create adaptable behavior and online learning capability that is essential for WMRs in varied terrains. As a result, the WMR can make decisions on-the-move, and it can be deployed for performing mission-critical tasks that involve unknown environments with mixed terrains.

As the proposed terrain classifier exhibits an adaptive learning behavior, it is often essential to redesign the classifier whenever there are variations of robot and environment with a new set of terrains. Hence, it becomes necessary to have a front-end to modify the robot environment and to study the performance of the proposed methodology and also to study its performance by changing the data, classifier design and performance measures. To this extent, a customized graphical user interface (GUI) is designed that serves as a tool to make changes in design and also to study the performance of the terrain classified.
3.4.2. Robot Wheel as Parametric Varying System

In the proposed work, the robot wheel interaction with the terrain is modeled as a parametric-varying system. The perturbations caused by the wheel-terrain interaction acts as a disturbance torque \( T_d \) against the motor torque \( T_m \). The resultant torque \( T_m - T_d \) drives the robot wheel, which will determine the robot wheel velocity \( V_w \). The motor torque is generated by the geared DC motor, which drives the robot wheel. The generated motor torque will be propositional to the supply voltage \( V_s \), which is regulated by a motor driver according to the duty ratio \( D\% \) as illustrated in Figure 3.15. Thus, the robot wheel velocity depends on the wheel-terrain interaction, which is characterized by the properties of the robot wheel such as threading, the material used, etc. and the terrain profile.

![Figure 3.15. Robot wheel system.](image_url)
Parametric variations are observed, as the same robot navigates across the various terrains due to changes in wheel-terrain interactions. Thus, a change in the territory can be assumed as an exogenous parameter \((\theta_{ex})\), which will have control over the system parameters and also influences the system output. The state space representation of the robot wheel interacting with the terrain can be considered as in Eq. (3.9).

In Eq. (3.9), \(u(n)\) is the input duty ratio, \((D)\) in percentage, \(y(n)\) is the output robot wheel velocity \((V_w)\) in cm/s and \(A, B, C\) are the system parameters. At steady state \((x(n) = 0)\), the system gain (friction coefficient) is given by (3.10), which shows that the system gain is a function of the terrain type \((K = f(\theta_{ex}))\).

\[
K = -C(\theta_{ex})A(\theta_{ex})^{-1}B(\theta_{ex})
\]  

(3.10)

3.4.3. Design of Terrain Classifier

The workflow in designing a terrain classifier consists of following steps: (i) data collection, (ii) pre-processing, (iii) modeling, and (iv) validation as shown in Figure 3.16. In the data collection step, the input-output data (duty ratio and wheel velocity) are collected from experiments. In the pre-processing step, the transients in the output wheel velocity are filtered by a transient filter after ensuring the input duty ratio is stable and within the operating range using a range filter and a steady state flag respectively. The filtered data are used to train an ANFIS that identifies the terrain type in the form of terrain number \((TN)\) in modeling step. The generated model provides a terrain perception mechanism that identifies terrains on-the-move. Finally, the perception mechanism is validated for its adaptability and robustness towards classifying the terrain in an unknown environment.
3.4.3.1. Terrain Profile

The friction-induced by wheel-terrain interaction characterizes the wheel slip and the robot mobility in the particular terrain. Therefore, it can be used to classify the terrain profiles. To classify terrains, this research uses the influence of wheel-terrain friction on steady-state gains \((K)\) of robot wheel. Four different terrain profiles: (1) cement concrete floor (CCF), (2) vinyl flex floor (VFF), (3) laminated wooden floor (LWF) and (4) vitrified tile floor (VTF), which commonly occurs in a typical indoor environment are considered for this study as shown in Figure 3.17. The CCF terrain offers a significant friction for wheel motion leading to a reduced system gain. The VTF has lesser wheel interaction followed by LWF and VFF, which will have a higher steady-state gain.

3.4.3.2. Experimental Data Generation

The objective of the proposed work is to estimate the terrain type using steady-state data that cause changes in system gain. An intelligent ANFIS classifier is designed to learn the intricate wheel-terrain interaction using the acquired input duty ratio \((D)\) and the robot wheel velocity \((V_w)\) across various terrains.

The data for training the ANFIS classifier is generated using experiments in which the duty ratio of the right wheel is changed from 0 to 100% in incremental steps of 1% whereas the left wheel is kept stationary. The duty ratio \((D)\) is maintained constant for three seconds to allow the wheels to reach steady-state, i.e., the rate of change of velocity tends to zero. The responses of the robot wheel for various terrains are acquired.
A closer inspection of the acquired data reveals the existence of a non-linear relationship due to complex wheel-terrain interaction. It is observed that there is a drop in wheel velocity for an increase in duty ratio beyond a point called ‘point of take-off’ (PTO). At this point, the torque generated by the robot right wheel becomes strong enough, and it is utilized to overcome the damping friction that exists between robot left wheel and ground. Figure 3.18 illustrates the acquired wheel velocity and the variations in PTO as observed for various terrains, which also simultaneously characterizes the terrain type.
3.4.3.3. Range Filter

The torque generated by the wheel motor requires overcoming not only the inertia of the robot wheel system but also the friction torque produced due to wheel-terrain interaction. Hence, at low duty ratio, the torque generated by the motor was not sufficient to overcome the damping friction of the terrains that in-turn depended on the terrain type. Thus, at the low range of duty ratio, the robot remains stationary, which leads to zero velocity ($V_w = 0$) and zero steady state gain ($K = 0$) as shown in Figure 3.19. The range of duty ratio ($0 < D < D_{min}$) at which the gain was approximately zero are called ‘dead zone’, and it is essential to eliminate these data for further processing. A range filter allows a valid data whenever the input duty ratio is above this dead zone. A range valid flag ($R_v$) is set when the input duty ratio is in the valid range as in Eq. (3.11).
\begin{equation}
R_v(n) = \begin{cases} 
1 & D_{min} < D < 100 \\
0 & \text{else}
\end{cases}
\end{equation}

### 3.4.3.4. Transient Filter

The proposed work uses the steady state behavior of the robot wheel to classify the terrain. For a given duty ratio, the robot wheel exhibits a transient behavior, and it reaches a steady state velocity after some time. This transient response of the robot wheel is eliminated by using a transient filter. Their time-varying behavior characterizes the transient data, and it can be filtered by using normalized acceleration \( nA_{rw} = \Delta V_{rw} / V_{rw} \). The acceleration has to be normalized since the rate change in velocity increase along with the increase in velocity. This is mainly due to motor vibrations at high speeds (higher duty ratio) as observed in Figure 3.18.

![Figure 3.18. Robot wheel velocity across various terrains.](image)
If the acceleration is within the tolerance limit \((nA_{tol})\), then a steady-state flag \((S_v)\) is set to indicate it as valid steady state data as in Eq. (3.12). Similarly, the transient data in the input duty ratio is also filtered by defining a steady-state flag \((S_D)\), which will be enabled when the input duty ratio is stable as in Eq. (3.13).

\[
S_v = \begin{cases} 
1 & |nA_{rw}| < nA_{tol} \\
0 & else
\end{cases} \tag{3.12}
\]

\[
S_D = \begin{cases} 
1 & \Delta D = 0 \\
0 & else
\end{cases} \tag{3.13}
\]

### 3.4.3.5. Steady State Gain

The steady state gains are calculated when all the inputs and outputs have reached steady state conditions within the operating range, which can be indicated by the flags. The steady gain is the ratio of output robot wheel velocity in cm/s and input duty ratio in % under steady-state conditions as given by Eq. (3.14).

\[
K = \left\{ \frac{V_w}{D} \middle| R_v = 1, S_v = 1, S_d = 1 \right\} \tag{3.14}
\]
3.4.3.6. Design of ANFIS Terrain Classifier

ANFIS combines the learning feature of the ANN with the decision making capability of FIS [72]. ANN tunes premise and consequent parameters of the FIS. The premise parameters determine the input membership functions that are used in fuzzification. The consequent parameters describe the output equations used in defuzzification. A rule base encapsulating the complex relationship between the inputs and outputs of the FIS is also generated using the knowledge learned by the ANN. It enables the FIS to take decisions on-the-move using the acquired inputs.

ANFIS classifier uses the training data to learn the complex nonlinear relationship between the input and output data. The proposed method uses steady state gain ($K$) and duty ratio ($D$) as inputs and terrain number ($TN$) as output for training the classifier. Once trained the ANFIS identifies the terrain in which the robot navigates using the instantaneous steady-state gain and duty ratio at run-time.

A first-order Takagi-Sugeno type FIS is employed for terrain classification with trapezoidal input membership function and linear equations as outputs as illustrated in Figure 3.20. The trapezoidal membership functions are selected for terrain classification since it has maximum membership degree for a range of input values around its center, which signifies a high degree of certainty. The degree degrades as the value deviates from the center, which indicates a lesser likelihood of that terrain type. For each input, five groups of membership function are chosen which produces optimum classification. It enables to create a rule base with 25 rules, which includes all the possible combination of membership functions. The classifier was trained using hybrid learning algorithm.
A hybrid learning algorithm [78] combines both the back-propagation algorithm (BPA) and, the least square method to tune the premise and consequent parameters of a FIS. The least-square method estimates the initial optimal parameters with the forward propagation of error and the BPA fine tunes them with the back-propagation of error. It ensures that optimal parameters are obtained for which the mean square error is minimal. Thus, the hybrid learning algorithm can train in a lesser number of iterations (epochs) as compared to BPA alone.

3.4.4. Design of GUI for Terrain Classifier

Robots intended to deploy for various applications, work in entirely different conditions and terrains. The environmental and other conditions change from one deployment space to another. To adapt the terrain classification algorithms to different grounds, researchers require a tool to simulate the performance of the proposed terrain identification algorithm for varied environments. Furthermore, to analyze the performance, graphs showing various facets of the terrain identification algorithm have to be presented to the user. In this investigation, a GUI is designed to address this requirement. It provides a user-friendly interactive tool that can be used by researchers to study the performance of the terrain classification algorithm in various environments. It provides an interactive tool for analyzing the classifier’s performance and its design modifications. It provides functionalities to generate training data, design/modify ANFIS classifier, and also performance evaluation tools.

The GUI for the proposed ANFIS terrain classifier is designed using MATLAB-GUIDE (Graphical User Interface Development Environment) toolbox, which can invoke the various key operating functional modules. This customized GUI is set to run on the
CPU of the robot such that it can have access to the robot’s hardware, in this case, the wheel odometer and the motor controller. During operation, it can be visualized from a remote computer connected to the robot’s CPU via remote connectivity using the wireless network.

The GUI primarily consists of five different panels namely, (i) ‘Flags’ panel, (ii) ‘ANFIS Classifier’ panel, (iii) ‘Actual Terrain’ Panel, (iv) ‘Robot Control’ Panel, and (v) ‘Actions’ Panel as shown in Figure 3.21.

The ‘Flags’ panel validates the acquired data, and it acts more like a gateway to allow a valid data to be used for terrain classification. The data are validated by three
flags explained in section 3.4.3, and a ‘Valid Data’ text will be flashed to notify the operator. Once the data is valid, the system evaluates the ANFIS classifier using this valid data to classify the terrain type.

![Figure 3.21. GUI for ANFIS terrain classifier.](image)

The ‘ANFIS classifier’ panel displays the run-time input ($D$ and $K$) and output data ($TN$) of the proposed classifier. It provides options to visualize the terrain type estimated by the ANFIS classifier. It is quite easy to validate the performance of the ANFIS classifier by comparing the estimated terrain type with the actual terrain type displayed in the ‘Actual terrain’ panel. Based on the visual observation of the robot’s navigating terrain, the operator has to give appropriate terrain type to the GUI to facilitate the training and testing operations of the terrain classifier.
The proposed GUI also offers options to redesign ANFIS classifier for a new set of terrains by loading them using ‘Load Terrain Types’ in the menu. The type of terrain is described by terrain number ($TN$) and the name of the terrain, which are stored in a Microsoft-Excel file (*.xls) to be loaded for redesigning the classifier for a new set of terrains.

It also provides options to control the robot using ‘Robot Control’ Panel. The robot can either be controlled manually or set to run autonomously with the predefined duty ratio. In manual mode, the motion of the robot is controlled by an operator using the dedicated keys (forward, reverse, turn left, turn right and halt). Once the robot is set to autonomous mode, the manual control keys are disabled, and the robot starts to navigate autonomously. In this mode, the robot uses its bumper switch to detect the obstacles and takes necessary action to avoid those obstacles. The ‘Classify Terrain’ button set the robot in motion with the given input duty ratio to generate data for terrain classification.

The ‘Action’ panel provides the three main functionalities (see Figure. 3.22) for ANFIS design as follows, (a) It provides options to generate training data. The training data is a set of input and output data acquired from the robot navigating in a known terrain. During training data generation, the input data such as input duty ratio, robot wheel velocity, and steady state gain are acquired from the robot. The output data (terrain number and terrain type) are obtained from the information provided by the operator. Once the data are generated, it provides options to save those data in a mat file (*.mat - data file format of MATLAB). (b) The ‘Design ANFIS classifier’ sub-panel gives a direct functionality to open MATLAB-ANFIS toolbox, where one can design and customize FIS to learn from the generated training data. (c) Finally, the designed ANFIS classifier
can be loaded and tested for its performance using the ‘Test ANFIS classifier’ sub-panel. The test results can be saved for further analysis using confusion matrix and for documentation.

![Figure 3.22. Sub-panels in ‘Action’ panel.](image)

The GUI can be invoked from the MATLAB command prompt, and the ‘START’ button connects CPU to the robot hardware. Once successfully connected, the operator has to select appropriate action from the ‘Action’ panel. During training and testing phase, the operator has to select the terrain in the ‘Actual Terrain’ panel to indicate the robot’s navigating terrain. Once the ANFIS classifier is designed and tested, it can be deployed in the robot for further operations. If the robot is deployed in a new environment with a new set of terrains, the operator has to load a new set of terrain numbers, and terrain types using the ‘Load Terrain Type’ option and the ANFIS classifier can be redesigned for the new environment.
3.4.5. Results and Discussion

The performance of the proposed ANFIS classifier is assessed under two different scenarios namely, (i) adaptive and (ii) robust testing in an unknown environment. The classification accuracy of the proposed algorithm is evaluated using confusion matrix, and the results are tabulated.

3.4.5.1. Adaptive Testing

In adaptive testing, the efficiency of the ANFIS classifier to classify the terrain when it encounters a terrain change is evaluated. A test environment with predefined terrain variations is designed. The robot is set to be in autonomous motion, and visual inspection observes the terrain in which the robot navigates in a particular interval of time. It corresponds to the actual terrain and is compared with the estimated terrain using the proposed classifier as shown in Figure 3.23. The results are tabulated in the form of confusion matrix, which indicates a 93.8% of classification accuracy. Thus, the proposed testing pattern evaluates the adaptive capability of the ANFIS classifier to classify a changing terrain environment.

3.4.5.2. Robust Testing

In robust testing, the robustness of the ANFIS classifier to indicate the same terrain in the presence of random motion and noisy sensor data is evaluated. The robot is set on random input duty ratio in the same terrain. To maintain uniformity, the input pattern is held constant, and the experiment is carried out across all the terrains. The results are described in the form of mean-variance graph as shown in Figure 3.24, which indicates the robustness of the proposed classifier. Further, a confusion matrix analysis shows a 94.2% of classification accuracy for the robust test pattern.
The proposed ANFIS classifier is also validated using data used during training, and the results are tabulated in the form of confusion matrix indicating an accuracy of 95.2% in terrain classification as shown in Table 3.5. In Table 3.5, the rows indicate the actual terrain type and the columns indicating the classified terrain type for various testing scenarios as discussed earlier.
Table 3.5. Performance validation using confusion matrix.

<table>
<thead>
<tr>
<th></th>
<th>Training Data [95.2%]</th>
<th>Adaptive Testing Data [93.8%]</th>
<th>Robust Testing Data [94.2%]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CCF</td>
<td>VFF</td>
<td>LWF</td>
</tr>
<tr>
<td>CCF</td>
<td>341</td>
<td>18</td>
<td>0</td>
</tr>
<tr>
<td>VFF</td>
<td>15</td>
<td>416</td>
<td>4</td>
</tr>
<tr>
<td>LWF</td>
<td>0</td>
<td>15</td>
<td>366</td>
</tr>
<tr>
<td>VTF</td>
<td>0</td>
<td>0</td>
<td>12</td>
</tr>
</tbody>
</table>

3.5. Summary

- Wheel slip is a major perturbation in the odometer, and it is essential to be compensated to improve the localization accuracy.
- Two methodologies have been adapted to compensate slip, one using intelligent self-learning NFS and another by using the stochastic multi-model approach.
- Use of intelligent NFS technique enables to capture the complex dynamics of wheel slip for compensation. Experimental results illustrate MSE of 0.0382 m and 0.0442 m during training and testing phase run respectively.
- Switching stochastic models are employed to compensate for the random nature of wheel slip, in various modes of robot motion, which can improve the mapping performance by 72.55% and 66.71% during rotation and translation motion respectively.
- Terrain dependent nature of wheel slip demands for terrain classification such that the slip can be compensated in specific to the terrain type.
Terrain types are identified by friction coefficient (gain) of the robot wheel as features and using ANFIS technique. Four terrain types that are common in the indoor environment are considered for this study, and the results show 93.8% and 94.2% of classification accuracy in adaptive and robust testing respectively.