Complex & Intelligent Systems (2024) 10:4175–4200
https://doi.org/10.1007/s40747-024-01367-6

ORIGINAL ARTICLE

HoloSLAM: a novel approach to virtual landmark-based SLAM for indoor environments

Elfituri S. Lahemer[1] · **Ahmad Rad**[1]

Received: 17 August 2023 / Accepted: 26 January 2024 / Published online: 4 March 2024

Abstract

In this paper, we present HoloSLAM which is a novel solution to landmark detection issues in the simultaneous localization and mapping (SLAM) problem in autonomous robot navigation. The approach integrates real and virtual worlds to create a novel mapping robotic environment employing a mixed-reality technique and a sensor, namely Microsoft HoloLens. The proposed methodology allows the robot to interact and communicate with its new environment in real-time and overcome the limitations of conventional landmark-based SLAMs by creating and placing some virtual landmarks in situations where real landmarks are scarce, non-existent, or hard to be detected. The proposed approach enhances the robot's perception and navigation capabilities in various robot environments. The overall process contributes to the robot's more accurate understanding of its environment; thus, enabling it to navigate with greater efficiency and effectiveness. In addition, the newly implemented HoloSLAM offers the option to guide the robot to a specific location eliminating the need for explicit navigation instructions. The open-source framework proposed in this paper can benefit the robotics community by providing a more reliable, realistic, and robust mapping solution. The experiments show that the Ellipsoidal-HoloSLAM system is accurate and effectively overcomes the limitations of conventional Ellipsoidal-SLAMs, providing a more precise and detailed mapping of the robot's environment.

Keywords EKF/Ellipsoidal landmark-based SLAM · Robotic mixed reality · Microsoft HoloLens · Unity3D · Virtual landmarks · HoloSLAM · Nao humanoid robot

Abbreviations

SLAM	Simultaneous localization and mapping
EKF	Extended Kalman filter
RXR	Robotic mixed reality
RAR	Robotic augmented reality
ESMF	Extended set-membership filter
IMUs	Inertial measurement units
RMS	Root mean square

✉ Ahmad Rad
arad@sfu.ca

Elfituri S. Lahemer
elahemer@sfu.ca

[1] Autonomous and Intelligent Systems Laboratory, School of Mechatronic Systems Engineering, Simon Fraser University, Surrey, BC, Canada

Introduction

Autonomous navigation is regarded as the key attribute of autonomous mobile robots, and simultaneous localization and mapping (SLAM) plays a critical role in its realization. The fundamental idea of SLAM is to concurrently construct or update a map of the robot's environment while estimating its pose using exteroceptive and proprioceptive sensors [1, 2]. Many successful implementations of SLAM have been reported in a wide range of environments, including indoor, outdoor, UAV, underwater, underground, and space [3–5]. Due to the complexity of these various environments, almost every approach for robotic navigation and SLAM solution relies, to some extent, on prior knowledge of the environment. Within this context, it is important to mention that each environment presents its own unique challenges, and there is no universal solution to SLAM. Consequently, SLAM solutions must be customized to the specific environment in which the robot operates to ensure precision and reliability in navigation [6, 7].

In landmark-based indoor environments, SLAM constructs a map by identifying distinct features within the environment, which can be either artificial or natural, and are typically referred to as landmarks [8, 9]. Landmarks in the context of SLAM can take the form of geometric features like lines or points, which are typically extracted from data collected by sensors such as laser range finders, sonar sensors, or visual features when cameras are used. Both artificial and natural landmarks share common characteristics; they are easily recognizable, distinguishable, abundant, and can be readily distinguished from their background within the environment [10]. A complete SLAM solution comprises several components, including landmark extraction, data association, state estimation, and state and landmark (map) update [11].

In natural landmark recognition, the landmarks are selected from unique features that exist naturally in the environment [12, 13]. Although the environment need not be altered for SLAM, having prior knowledge of the surroundings is crucial. However, methods relying on such prior knowledge can be less reliable, as the distinct features they depend on may change or become less distinguishable over time. Nonetheless, these methods have the advantage of not requiring the preparation or installation of additional landmarks, making them more practical in specific situations. Indoor settings may have passive natural landmarks such as doors, windows, and ceiling lights, while outdoor environments can have landmarks such as roads, trees, and traffic signs [14, 15]. Information from these landmarks can also be extracted using machine learning and deep learning techniques [16]. Artificial landmarks are intentionally created, designed, or modified to be easily recognizable by a robot's sensors. These landmarks are added or placed in specific or random locations in the environment with the specific purpose of aiding in the robot's navigation, localization, or mapping tasks. The robot's sensors are expected to detect and recognize these artificial landmarks to improve its understanding of its surroundings and to determine its position and orientation accurately [8, 17]. The robot is specifically programmed or equipped to detect and interpret these landmarks, making them an integral part of the robotic system's sensor data and mapping processes. Predetermined artificial landmarks offer benefits in autonomous navigation due to their high visibility, ease of distinction from other objects, and the ability to convey additional encoded information to the robot. This makes them valuable for cost-effective and reliable navigation, especially in complex environments where natural landmarks may be unreliable or scarce.

In order for a landmark-based SLAM system to operate effectively, the landmarks must be both present in the environment and within the range of the robot's sensors. The robot requires mobility to observe these landmarks from various positions and angles [8, 15]. The presence of these landmarks is not always guaranteed, and their absence poses a challenge to the system's ability to accurately estimate the robot's position and map the surroundings [18]. The accuracy of position estimation in landmark-based SLAM systems is often dependent on the distance and orientation between the robot and the landmark. When the robot is further away from the landmark, the features of the landmark may appear smaller in the sensor data, which can make it harder to accurately detect and track. Similarly, when the orientation between the robot and the landmark is not optimal, the features of the landmark may not be easily distinguishable, which can also lead to lower accuracy in position estimation. As the robot approaches a landmark, the characteristic of the landmark becomes more apparent and easier to identify and track, resulting in improved accuracy in the robot's position estimation [18–20]. Furthermore, the precision of the landmark detection system is paramount. If the detection system is unreliable or the probability of detecting a landmark is less than one, the SLAM system may generate imperfect landmarks or fail to identify certain ones. Environmental factors such as noise, illumination changes, and other factors can also influence sensor performance, impacting system accuracy and reliability. This introduces inaccuracies in mapping and position estimation, resulting in errors in navigation and other tasks.

To guarantee the stability and reliability of a landmark-based SLAM solution, the detection of at least one landmark in every observation step is indispensable. Developing a landmark-based SLAM solution that can effectively utilize multiple types of landmarks poses a considerable challenge [11, 19]. Many existing solutions addressing this issue often demand higher computational costs and the incorporation of additional sensors to achieve optimal performance. It is important to note that real-time modifications, such as adding or removing landmarks, are not feasible in this context.

Landmark-based SLAM solutions that often utilize extended Kalman filters (EKFs) to estimate the state of the robot and landmarks employs a single Gaussian to represent the structure of each state [21–24]. In these approaches, the environment map is represented by the respective positions of separate landmarks or distinctive features. Therefore, the robot is enabled to get absolute pose estimates of the environment [25, 26]. These landmarks are then frequently re-observed, their locations are updated until the mapping process is complete [27].The state vector in these solutions includes only the landmark position and the robot pose suggesting that SLAM by itself does not provide additional information about the environment [26]. This implies that the landmark-based SLAM alone may not provide enough information to fully understand the environment. A key limitation of EKF-based landmark mapping is its vulnerability to false data associations, where measurements are incorrectly linked to landmarks [25]. This problem is more significant

when data association methods lack correlation consideration, and it escalates as the number of landmarks increases, imposing a higher computational burden [26].

To address the Gaussian assumption of EKF, a set-theoretic approach was introduced in Ref. [28]. It assumes bounded errors to achieve information fusion through set intersections. This method imposes strict noise and state estimate bounds. It has been extended to a mixed stochastic/set-membership framework in Ref. [26], where ellipsoidal approximations of robot poses are used to describe probabilistic uncertainties. While Ellipsoidal-SLAM provides more accurate landmark position estimates than EKF, it still depends on the availability of numerous real, unique, predefined, and distinctive landmarks during each update. Complete resolution of data association remains a challenge. Furthermore, the state does not provide additional information about landmark details.

Robotic augmented reality (RAR) [26] is integrated with EKF/Ellipsoidal-SLAM to enhance performance, reduce computational load, facilitate landmark identification, and streamline data association. Augmented reality (AR) [29] fundamentally blends real-world views with computer-generated graphics to enhance our perception of the environment.

Mixed reality (MR) offers additional benefits to augmented reality by overlaying virtual objects or landmarks onto the real physical environment [30]. This could be a robot environment-changing tool for all phases of environment design, interaction, and communication. Integrating the Microsoft HoloLens or any other mixed-reality devices with the robot's real-world sites will offer new and alternative possibilities for the robot to communicate and interact with its environment [31]. The HoloLens is then assumed to achieve this using a combination of common methods and techniques to solve the SLAM problem where the construction and updating of the robot environment is done while simultaneously keeping track of the robot's location within this environment [32]. Mixed reality using HoloLens falls between AR and VR, where one can experience the virtual objects merged with the physical objects, not by looking through the transparent lenses of a HoloLens headset only but also by interacting with the virtual object using a natural interface.

The motivation of this study is to address some of the abovementioned problems related to landmarks in the SLAM process. We introduce HoloSLAM that essentially leverages mixed reality to provide a set of virtual landmarks for the robot to be used for navigation. Using robotic mixed reality (RXR) and HoloSLAM allows the robot to navigate and map its surroundings using a set of virtual landmarks and voice commands, without the need for predefined multiple landmark detectors and without any human help or intervention. This enables the robot to independently define, select, and position a set of virtual landmarks within its environment. HoloSLAM also empowers the robot to experience mixed-reality techniques, like how humans do, using its voice commands and without relying on object detectors or predefined models of the real environment. These capabilities are achieved without any human assistance or intervention. HoloSLAM effectively addresses the data association problem through the utilization of RXR, akin to a robotic augmented reality (RAR) approach. Indeed, our implemented HoloSLAM method proficiently addresses diverse challenges associated with landmarks in landmark-based SLAM, including scarcity, clarity, presence, absence, detection issues, and data association problems. The robot simulates a human-like mixed-reality experience, placing/removing virtual landmarks, adjusting the map in real-time via voice commands, and navigating without explicit instructions. HoloSLAM is designed to function with minimal sensor requirements and can operate within landmark-free SLAM constraints. Moreover, HoloSLAM is compatible with diverse robot platforms. In addition, we intend to publish an open-source framework, making it accessible to the wider robotics community, thus fostering collaboration and facilitating advancements in the field.

We validate our implemented system (HoloSLAM) in real-time experiments in the AISL lab with Nao humanoid robot. Therefore, our objective is to develop a SLAM system on Nao robot [32] or any other robots and address the challenges associated with landmark detection in the environment. This system aims to enhance the robot's perception of both its virtual and real surroundings.

The paper is organized as follows: we present a selected literature review of the related algorithms in the section "Related studies". We then provide an in-depth overview of the structure of the proposed SLAM. In the section "Experimental results", we will include experimental studies and discuss the merits of the proposed architecture. We conclude the paper with general remarks in the section "Conclusions".

Related studies

Navigation in unknown or partially known environments often requires a SLAM solution whereby the autonomous mobile robot maps its environment while concurrently localizing itself [27]. SLAM research and its solutions can be classified into numerous categories based on various characteristics. Each solution or method is distinguished by the sort of map they generate, or the sensors robots employ or by the sort of mathematical algorithm that is utilized to solve the SLAM problem. The robot platform being used to solve SLAM problem is another challenge to be considered. In the context of probabilistic robotics, there have been several

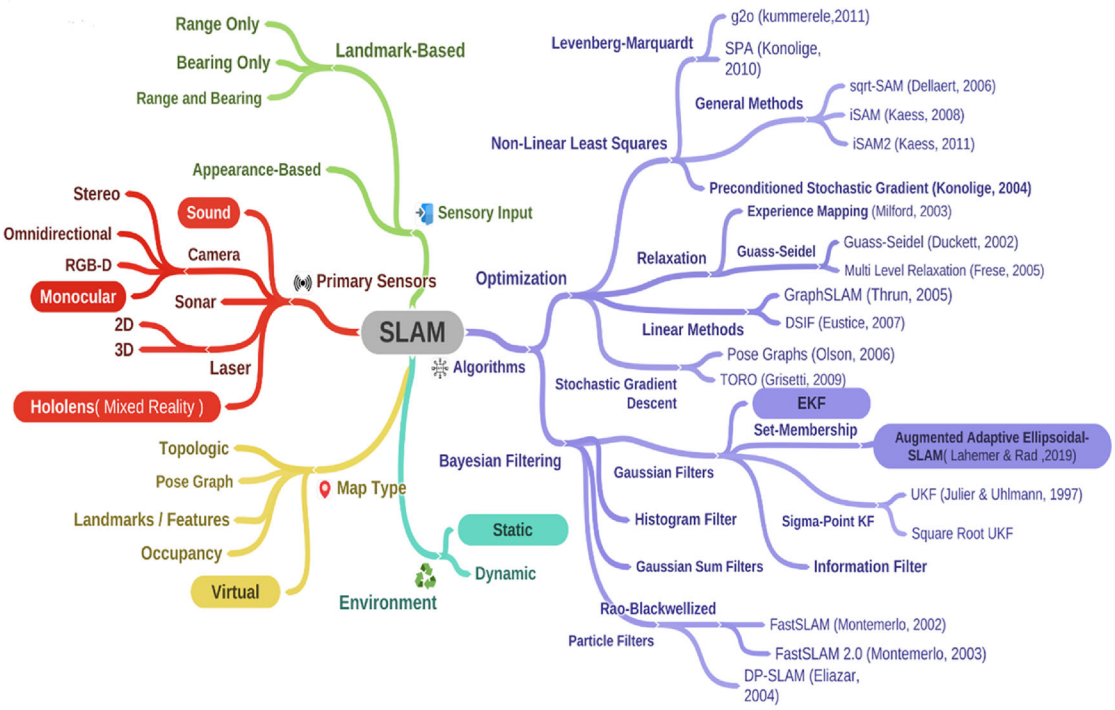


Fig. 1 SLAM solution and research classifications

landmark-based SLAM approaches for various environments [11, 12, 27, 33]. Figure 1 shows the SLAM solutions and its research classifications, and this paper primarily focuses on the highlighted areas.

The structure of the simultaneous location and mapping problem and the coining of its acronym SLAM was first proposed in a seminal paper in 1996 [34]. The environment was modeled by a finite number of distinguished landmarks that were visible and re-observable by robot sensors in the environment. Their general idea was to store robot poses and landmark locations in a combined, Gaussian distributed vector and perform EKF to update the estimation at each time step [35]. The state vector estimated by this SLAM solution includes only the positions of landmarks and robot poses, without providing additional information about the specific details of the landmarks. For the EKF-SLAM, the cost grows in a quadratic manner with the number of landmarks. The performance of EKF-SLAM algorithm deteriorates rapidly when observed features are associated with wrong landmarks. Even a single incorrect data association may have severe consequences for the map and hence the localization of the robot [36, 37].

All probabilistic feature/landmark-based Online/Full SLAM solutions presented in Refs. [38–41] include Kalman filter and its variants such as EKF, information filter, particle filter, and graph optimization solutions use predefined landmarks to estimate the robot current state (or robot path) and create a map. These solutions necessitate multiple landmarks that must be observed from various positions and angles. These solutions can fail when no landmarks are present or when the robot cannot detect any using its sensors, leading to convergence problems. In intricate, large-scale applications, placing hundreds of markers throughout the environment poses a significant challenge. Indeed, some of these solutions suffer from the Gaussian distribution assumption of uncertainties, while this assumption can simplify the mathematical formulation and computation, it may not always hold in real-world scenarios where the uncertainties may be non-Gaussian, multi-modal, or even nonlinear.

To cope with Gaussian noise premise, a set-membership approach was introduced in Ref. [42] under the assumption of bounded errors to obtain information in which a probabilistic uncertainty description is associated to ellipsoidal approximations of the admissible robot poses.

The concept of robotic augmented reality (RAR) is introduced and used in Ref. [26] to improve the SLAM performance. The authors implemented augmented adaptive Ellipsoidal-SLAM which used RAR technique to provide more information about the landmarks in the environment. This successfully solved the data association problem and reduced the computational cost problem. However, this solution still requires sufficiently unique, plentiful, predetermined, and distinct real artificial landmarks (NaoMarkers) to exist in the robot environment in every update step.

Other SLAM solutions that are not based on the landmark's detection approaches assume a much larger amount of unidentifiable information, as provided by sensors such as laser and sonar-based range finders. Although these approaches consider a much larger dataset, the computation time of these approaches resembles that of landmark-based approaches. An example of a non-landmark-based SLAM approaches is DP-SLAM [43] or the GMapping system [44, 45].

DP-SLAM tackles the data association challenge by storing multiple intricate maps instead of relying on sparse landmarks. This approach combines data association with localization, allowing the authors to assert that DP-SLAM does not make any assumptions about landmarks. Through the localization of these multiple detailed maps, DP-SLAM effectively generates what can be referred to as "invisible landmarks". Consequently, the DP-SLAM process leans more toward localization rather than a comprehensive SLAM solution.

Our approach (HoloSLAM) provides a novel and new way to define SLAM mapping method using robotic mixed reality (RMR) which gives the robot itself the ability to provide and place a set of virtual landmarks in its environment and enables the Nao robot or any other robot to experience the mixed-reality technique in a similar way to the human using its voice natural language understand commands and without being equipped with object detectors for multiple novel objects or a predefined real environment model without any human help and intervention. The robot will also be able to interact with these virtual landmarks, modify, delete and place any of these virtual landmarks using his voice command. The approach is also optionally able to instruct the robot to navigate to a specific position by saying "go to the landmark" rather than giving an explicit navigation instruction. HoloSLAM can deal with the data association problem using RMR in a similar way to a RAR. HoloSLAM can also deal with any landmark SLAM issues including and not limited to its shortage, clarity, presence, absence, and extraction. In addition, it can work with limited sensor requirements and with landmark-free SLAM constraints.

Methodology and implementation

SLAM problem and solutions

SLAM can be viewed as an estimation problem for an uncertain dynamic system by taking into account a description of the environment based on landmarks and noisy measurements [26]. Let us consider the scenario of an autonomous robot (Nao in this case) navigating through an unfamiliar environment and perceiving multiple stationary landmarks using its built-in sensor (Fig. 2). At time **k**, we have

$$\begin{cases} \boldsymbol{x_t}: & \text{the vector describing the position and orientation of the robot.} \\ \boldsymbol{u_k}: & \text{the vector, applied at time } k-1 \text{ to move the vehicle to } x_k \text{ at time} k. \\ \boldsymbol{m_i}: & \text{the vector describingthe position of the } i\text{th stationary landmark.} \\ \boldsymbol{z_{ik}}: & \text{the observation, taken from the robot, of the } i\text{th landmark at time } k. \end{cases}$$

The robot requires a control signal $\boldsymbol{u_t}\{\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u_k}\}$ to be able to move from position $\boldsymbol{x_{t-1}}$ to $\boldsymbol{x_t}$ using the IMU (Inertial Measurement Unit) which has some uncertainty in its measurement, and the robot takes some observation $\boldsymbol{z_t}$ using its external sensors, e.g., laser. The robot poses $\boldsymbol{x_t}$ and/or the landmark positions $\boldsymbol{m_i}$ are concurrently estimated. The goal in every iteration is to find the posterior data which refers to the estimated data of robot position (or the whole trajectory) and all landmarks' positions together and reduce the measurement and observation errors. It can be represented in one vector the environment map containing a list of objects.

In the 2D case, the state $\boldsymbol{s_t}$ at time **t** can be represented by the following column vector:

$$\mathbf{s_t} = \left(\underbrace{\mathbf{p_{x,t}}, \mathbf{p_{y,t}}, \mathbf{p_{\theta,t}}}_{robot'spose}, \underbrace{\mathbf{m_{x,1}}, \mathbf{m_{y,1}},}_{landmark1}, \underbrace{\mathbf{m_{x,2}}, \mathbf{m_{y,2}},}_{landmark2} \cdots\cdots\cdots \underbrace{\mathbf{m_{x,N-1}}, \mathbf{m_{y,N-1}},}_{landmarkN-1} \underbrace{\mathbf{m_{x,N}}, \mathbf{m_{y,N}}}_{landmarkN} \right)^{\mathbf{T}} \quad (1)$$

where $\boldsymbol{x_t} = (\boldsymbol{p_{x,t}}, \boldsymbol{p_{y,t}}, \boldsymbol{p_{\theta,t}},)$ denotes the robot's coordinate and $\boldsymbol{m_{x,i}}, \boldsymbol{m_{y,i}},$ are the coordinates of the $\boldsymbol{m_i} i = 1, \ldots, N$th landmark in $\boldsymbol{x}$–$\boldsymbol{y}$ plane.

The uncertainty in model and measurement systems is handled by probabilistic techniques, one of the dominant paradigms for algorithm design in robotics navigation. These approaches represent uncertainty and ambiguity through explicit "acquired belief" using probability theory and form robust control choices relative to the remaining uncertainty in the model. Standard solutions are provided by the extended Kalman filter (EKF) or other probabilistic techniques and its latest solution adaptive augmented Ellipsoidal-SLAM [26, 46].

Fig. 2 Nao robot with HoloLens in indoor environment performing SLAM

SLAM was first solved by EKF-SLAM which is still regarded as one of the most influential and widely used implementations [47]. The EKF essentially linearizes the nonlinear functions around the current state which is accomplished by Taylor expansion technique. After this linearization process, classical Kalman filter is employed to estimate states.

EKF-SLAM estimates the state from a series of noisy measurements (movements and observations), where the noise distribution is assumed to be Gaussian. More details on EKF-SLAM implementation on Nao robot are available in Refs. [26, 48].

In general, the EKF-SLAM is done in two steps: prediction and correction [35, 49]. In the SLAM problem, during the prediction step the state at time is updated according to the motion model:

$$\overline{\mu_t} = \mathbf{f}\left(\mathbf{u_t}, \boldsymbol{\mu}_{\mathbf{i}-1}\right) \tag{2}$$

$$\boldsymbol{\Sigma}_t = \mathbf{F_t}\boldsymbol{\Sigma}_{t-1}\mathbf{F_t^T} + \mathbf{Q_t} \tag{3}$$

where $\boldsymbol{\mu}_{i-1}$ and $\boldsymbol{\Sigma}_{t-1}$ are, respectively, the mean and covariance of the state at time $t-1$. $\boldsymbol{F}_t$ is the $n \times n$ Jacobian matrix of the function $\boldsymbol{f}$ (n is the dimension of the state). The Jacobian usually depends on $\boldsymbol{u}_t$ and $\boldsymbol{\mu}_{t-1}$:

$$\mathbf{F_t} = \nabla_{\mathbf{s_{t-1}}}\mathbf{f}(\mathbf{u_t}, \mathbf{s_{t-1}})|_{\mathbf{s_{t-1}}=\boldsymbol{\mu}_{\mathbf{t-1}}, \mathbf{s_t}=\boldsymbol{\mu}_{\mathbf{t}}}$$

and $\boldsymbol{Q}_t$ covariance matrix that models the uncertain in the state transition as Gaussian noise with zero mean $\sim \boldsymbol{N}$ $(\mathbf{0}, \boldsymbol{Q}_t)$.

During the correction step, for every observation z_t associated with the landmark $\boldsymbol{m}_i$, it is computed an expected observation $\widehat{z}_t = \boldsymbol{h}\left(\overline{\mu_t}, \boldsymbol{m}_i\right)$ and a corresponding Kalman gain (5) that specifies the degree to which the incoming observation corrects the current state estimation (Eqs. 6 and 7):

$$\mathbf{S_t} = \mathbf{H_t}\boldsymbol{\Sigma}_\mathbf{t}^- \mathbf{H_t^T} + \mathbf{R_t} \tag{4}$$

$$\mathbf{K_t} = \boldsymbol{\Sigma}_\mathbf{t}^- \mathbf{H_t^T}\mathbf{S_t^{-1}} \tag{5}$$

$$\boldsymbol{\mu}_\mathbf{t} = \overline{\mu_t} + \mathbf{K_t}(\mathbf{z_t} - \mathbf{h}(\overline{\mu_t}, \mathbf{m_i})) \tag{6}$$

$$\boldsymbol{\Sigma}_\mathbf{t} = (1 - \mathbf{K_t}\mathbf{H_t})\boldsymbol{\Sigma}_\mathbf{t}^- \tag{7}$$

where $\boldsymbol{h}$ is a function that maps the current state in an expected observation $\widehat{z}_t$ given the landmark $\boldsymbol{m}_i$ associated to $\widehat{z}_t$, $\boldsymbol{R}_t$ covariance matrix that models the observation noise as Gaussian noise with zero mean $\sim \boldsymbol{N}(\mathbf{0}, \boldsymbol{R}_t)$ and where $\boldsymbol{H}_t$ is the Jacobian of the function $\boldsymbol{h}$:

$$\mathbf{H_t} = \nabla_{\mathbf{s}t}\mathbf{h}(\mathbf{u_t}, \mathbf{m_i})|_{\mathbf{s_{t-1}}=\overline{\mu_t}}, \mathbf{m_i} = \mathbf{m_i} \tag{8}$$

The standard formulation of the EKF-SLAM solution is not robust and is prone to incorrect association of observations to landmarks: an accurate data association is then desirable [50]. EKF-SLAM requires analytic models of the vehicle motion and observations, it makes a few assumptions which are often violated in practice, and it will fail whenever data association fails. In addition, measurements for incorrectly identified landmarks would still be integrated by the EKF, producing an incorrect map.

Ellipsoidal set-membership filter method for SLAM (Ellipsoidal-SLAM). In this approach, the nonlinear dynamics are linearized about the current estimate in a manner that is like the EKF. The remaining terms are then bounded, and they are incorporated into the algorithm as additions to the process or sensor noise bounds [28]. It is also computationally

efficient for online recursive implementation. The new algorithm is termed the extended set-membership filter (ESMF) [42]. Figure 3 shows the modular process of ESMF. Readers may refer to Ref. [26] for more details on Ellipsoidal-SLAM and its implantation on Nao robot.

The adaptive ellipsoidal-SLAM algorithm can be summarized as follows [26]:

Start
Initialization -
$\hat{x}_{1,1} = 0 \;\; P_{k,k} = 0$
Get Observation – $k_z = 1$
$\mathbf{z_1}$ =**get_observations**, Looking for QRcodes.
while not_stop
Prediction Step – (4) Check for safe distance to move by sonar. Move command.
No safe distance. Turn 180 degree

$$[x_{K+1},\ P_{k+1,k}] = \text{prediction}\,(x_{k/k},\ P_{k,k},\ u_k)$$

$$\beta_{Q_k} = \frac{\sqrt{\mathrm{Tr}(Q_k)}}{\sqrt{\mathrm{Tr}(Q_k)} + \sqrt{\mathrm{Tr}(Q_k)}}$$

$$\hat{Q}_k = \frac{\bar{Q}_k}{1-\beta_{Q_k}} + \frac{Q_k}{\beta_{Q_k}} \quad , \beta_{Q_k} \in (0,1)$$

$$\beta_{Q_k} = \frac{\sqrt{\mathrm{Tr}(\hat{Q}_k)}}{\sqrt{\mathrm{Tr}(\emptyset_k P_k \emptyset_k^T)} + \sqrt{\mathrm{Tr}(\hat{Q}_k)}}$$

$$P_{K+1,k} = \emptyset_k \frac{P_k}{1-\beta_k} \emptyset_k^T + \frac{\hat{Q}_k}{\beta_k} \quad , \ \beta_k \in (0,1)$$

$$\emptyset_k = \partial f(x_k) / \partial x_k \Big|_{x_k = \hat{x}_k}$$

Get Observation (5)- Looking for QR codes by turning head by 30 degrees.
If Nao find QR codes to robot frame
Data Association $(z_k, \hat{x}_{k/k},\ P_{k+1,k})$
Correction _Step - $(x_{k/k},\ P_{k+1,k},\ z_k)$

$$w_k = H_{k+1} \frac{P_{k+1,k}}{1-\rho_k} H_{k+1}^T + \frac{\hat{R}_{k+1}}{\rho_k} \rho_k \in (0,1)$$

$$k_{k+1} = \frac{P_{k+1,k}}{1-\rho_k} H_{k+1}^T w_k^{-1}$$

$$\hat{x}_{k+1} = \hat{x}_{k+1,k} + k_{k+1}[y_{k+1} - h(x_{k+1})]$$

$$\bar{P}_{k+1,k} = \frac{P_{k+1,k}}{1-\rho_k} - \frac{P_{k+1,k}}{1-\rho_k} H_{k+1}^T w_k^{-1} H_{k+1} \frac{P_{k+1,k}}{1-\rho_k}$$

$$\rho_k = \frac{\sqrt{\mathrm{Tr}(R_{k+1})}}{\sqrt{\mathrm{Tr}(H_{k+1} P_{k+1} H_{k+1}^T)} + \sqrt{\mathrm{Tr}(R_{k+1})}}$$

Map_Step- $(x_{k+1/k+i},\ P_{k+1,k+1},\ z_k)$ Add new QR code to the map
No QR codes –Turn Naoby 180 degree and go to step(5)

$$k_z = k_z + 1$$

Check if iteration numbers are achieved.
NoGo to step (4)
K=K+1
End

where $\boldsymbol{\rho_k}$ and $\boldsymbol{\beta_k}$ are filter parameters to be chosen online to minimize the ellipse.

Typically, during a SLAM update, the goal is to decrease uncertainty and enhance the overall state estimate, especially when re-observing a landmark. If no

landmark is detected, the pose estimate's accuracy diminishes due to actuator uncertainty, leading to error accumulation reflected in increased values in the covariance matrix. This trend continues until a new landmark is detected, resulting in a significant reduction in uncertainties. The success of any SLAM algorithm relies on detecting landmarks to mitigate IMU errors, and failure occurs when real landmarks are not found or are inaccurately detected by the robot sensors. Moreover, the state vector includes only landmark positions and robot poses, lacking detailed information about the unique properties of the landmarks. SLAM, on its own, does not offer specific details about the landmarks, which may result in false positives during data association. Accurate data association and landmark identification are crucial to prevent convergence towards an incorrect SLAM solution. Inaccurate data association increases the computational cost in obtained EKF/Ellipsoidal-SLAMs.

The robotic augmented reality (RAR) technique, as applied in Ref. [26] enhances map representation by incorporating additional information about landmarks, addressing landmark identification and recognition, and simplifying the data association issue. The adaptive augmented Ellipsoidal-SLAM implementation fails when no landmarks (NaoMarkers) are detected during the measurement stage.

Robotic mixed reality (RXR) provides extra advantages over augmented reality by superimposing virtual landmarks (holograms) onto the actual physical environment. Moreover, these virtual landmarks can be strategically positioned to remain visible to the robot's sensors, even in challenging settings with limited visibility or intricate structures. This makes RXR a compelling solution for addressing the SLAM problem in diverse applications and environments, particularly when real landmarks are absent or difficult to detect in the robot's surroundings. The project employs robotic mixed reality (RXR) to create real-time virtual landmarks that enhance EKF/Ellipsoidal-SLAM performance and address SLAM landmark-related issues. Mixed reality, a recent technology integrating virtual computer-generated data (includes 2D/3D objects, voices, texts, images, or any graphic content, often referred to as holograms) into the real-time environment, is utilized to add virtual landmarks and augment the robot's environmental perception.

In the following section, we briefly introduce mixed reality, including augmented reality, virtual reality, and the concept of robotic mixed reality, emphasizing its advantages for autonomous robot navigation. This study has explored Microsoft's HoloLens functionality, and developed a customized hologram for display on HoloLens, to be mounted into Nao's head.

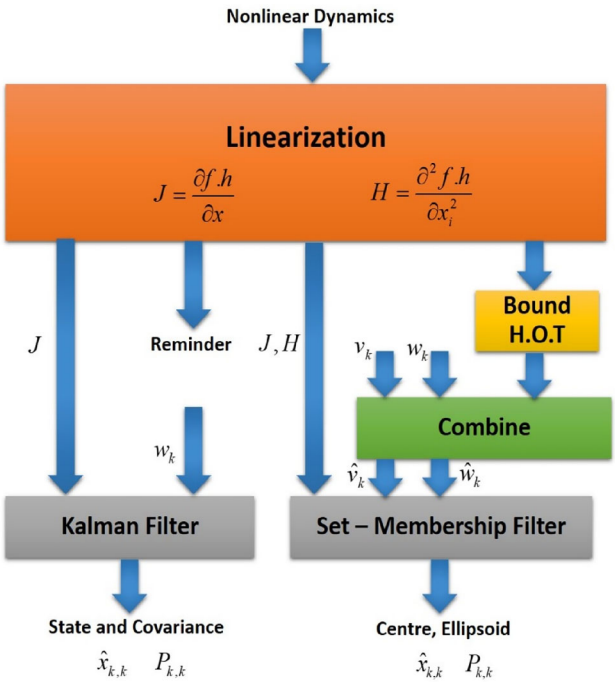


Fig. 3 Simplified graphical representation of the EKF and the nonlinear set-membership filter

Robotic mixed reality (RXR)

Augmented, virtual and mixed realities are emerging areas and are expected to have major impact in many areas including robotics [51, 52]. Integrating mixed reality with robotics holds promise for creating innovative applications, although current research mainly focuses on human–robot interaction (HRI) rather than autonomous navigation. This study pioneers the integration of robotic mixed-reality technology with EKF/Ellipsoidal-SLAMs algorithms, offering a real-time solution to SLAM landmark-related challenges.

The word mixed reality comes from the research paper by Paul Milgram and Fumio Kishino entitled, Taxonomy of Mixed-Reality Visual Displays published in 1994 [53]. In this publication, the term was introduced in the context of a segment of the virtuality continuum, later referred to as the reality–virtuality (RV) continuum. Figure 4 shows the definition of mixed reality within the context of the real–virtual continuum.

This spectrum spans environments from the entirely real world to fully virtual spaces. Mixed reality is the integration or interaction between these two realms. A straightforward interpretation of mixed reality is the presentation of real-world and virtual objects together in a single display, covering the range between the virtuality continuum's extremes [54]. Since its publication, the use of mixed reality has expanded beyond displays to include environmental

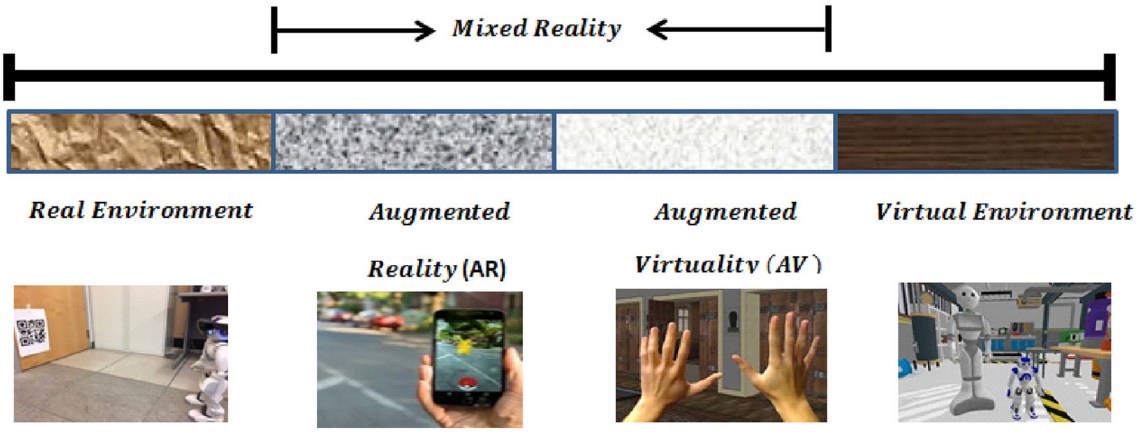


Fig. 4 Definition of mixed reality within the context of the real–virtual continuum

input, spatial sound, and location. The convergence of computer processing, human/robot input, and environmental input opens up the potential to create authentic mixed-reality experiences, enabling movement in the physical world to correspond with movement in the digital realm.

Mixed reality involves integrating virtual, computer-generated data into a real-time environment [55]. Typically, this added data includes images or sound, but it can extend to other forms such as video or tactile information. Serving as a combination of augmented reality and virtual reality, mixed reality merges digital data with the real world, allowing for interaction between the two [56]. Its distinctive trait lies in the capability to make digital data interact with the real world.

Robotic mixed reality (RXR) and its applications introduce novel ways to perceive and navigate the robot's surroundings using holograms and virtual data like images and sound. To process both digital data and real-world information, devices necessitate powerful CPUs and graphical processors (GPUs) for digital rendering and data generation. Various display devices, whether lenses or physical screens, are essential to showcase the generated digital information. Various devices facilitate the creation of immersive environments. Regarding augmented reality (AR), commonly utilized devices include Microsoft HoloLens, Magic Leap One, Epson Moverio, and Google Glass. In the realm of virtual reality (VR), popular choices include HTC Vive, Oculus Quest, Valve Index, and Sony PlayStation VR. In 2016, Microsoft released HoloLens, a head-mounted display (HMD) for mixed-reality production [31]. In addition to Microsoft, other companies have developed HMD display devices for augmented and mixed reality and experience, as well as various smart glasses [57].

Robotic mixed reality (RXR) can be defined as a combination of computer processing, robot input, and robot environmental input as illustrated in Fig. 5. Robot input can occur through original means that could include gestures, touch, and voice.

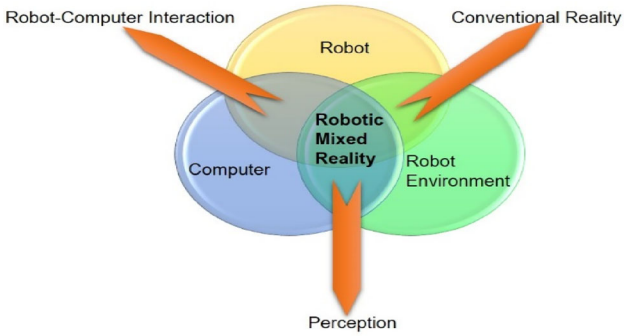


Fig. 5 Robotic mixed-reality (RXR) diagram

The aim of the robotic mixed-reality technology in this project is to create virtual landmarks (holograms) in real-time to address SLAM landmark-related issues. This technology enables the robot to interact with these landmarks, experience mixed reality like humans, enhance robotic mapping, and improve overall SLAM performance for autonomous navigation. A virtual landmark denotes a digital or virtual entity functioning as a distinct and recognizable point or feature in an augmented or mixed-reality setting. This encompasses various items, such as a 3D model, holographic representation, or interactive element strategically positioned in the physical space of the robot. These digital artifacts enhance the robot's environment by seamlessly coexisting with the real world. Figure 6 shows the new robot environment including some new virtual landmarks.

The new state vector of our SLAM can be represented by the following:

$$\mathbf{s_t} = \Bigg(\underbrace{\mathbf{p_{x,t}}, \mathbf{p_{y,t}}, \mathbf{p_{\theta,t}}}_{robot'spose}, \underbrace{\mathbf{m_{x,1}}, \mathbf{m_{y,1}}}_{landmark1}, \underbrace{\mathbf{m_{x,2}}, \mathbf{m_{y,2}}}_{landmark2}, \cdots,$$

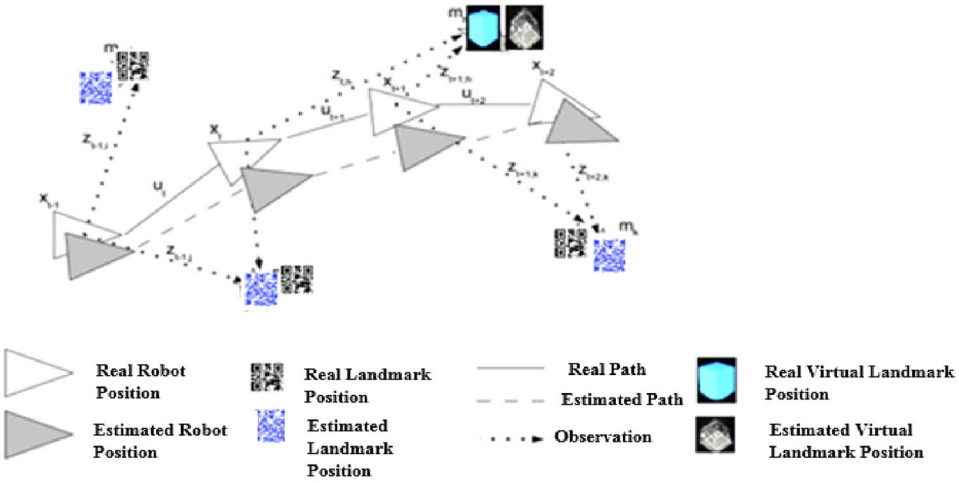


Fig. 6 Mixed-reality robot environment includes some virtual landmarks

$$\ldots \underbrace{\boldsymbol{mv}_{\mathbf{x},1},\boldsymbol{mv}_{\mathbf{y},1},}_{virtuallandmark}\ \underbrace{\mathbf{m}_{\mathbf{x},\mathrm{N}},\mathbf{m}_{\mathbf{y},\mathrm{N}}}_{landmarkN}\Bigg)^{\mathbf{T}}. \tag{9}$$

This new state vector contains new virtual landmark IDs ($\boldsymbol{mv}_{y,I},\ldots\boldsymbol{mv}_{x,I},$) which can solve any landmark issues in the robot environment. This enhancement facilitates the robot's path navigation and enables seamless continuation of SLAM operations.

In this work, we have combined robotic mixed reality via Microsoft HoloLens with Ellipsoidal-SLAM algorithms to enhance SLAM performance and resolve landmark issues. HoloLens employs various methods typical for SLAM problem-solving, involving the construction and updating of the environment while tracking the robot's location within it. Next section gives more details about Microsoft HoloLens.

Microsoft HoloLens

In 2016, Microsoft launched HoloLens as the first personal device to use and experience the mixed-reality digital environment. HoloLens is on the market as a developer, whose usage and hardware performance are expected to increase in future. The following section gives a more detailed description about HoloLens' structure, techniques, future capabilities, and ways to display and create holograms for HoloLens use.

HoloLens technology and components Microsoft HoloLens operates on a customized version of Windows 10. This integration enables HoloLens to support Cortana, Microsoft's artificial intelligence, functioning as a voice recognition and task assistant. The device comprises a transparent visor housing main cameras and sensors for real-world virtualization. The transparent visor and screens allow mixed-reality data to be seamlessly displayed in the user's physical environment, a few meters away [58]. The hologram display relies on light refraction as the HoloLens camera transmits light to the lenses. The light then bends through the lenses into the eyes, creating holograms. Figure 7 depicts the overall structure of HoloLens, highlighting the positions of the visor and displays on the left. In addition, it illustrates the mounting of HoloLens on the wheeled robot. HoloLens cameras serve functions like brightness control, video recording, and spatial mapping. The device features a depth camera to read space depth elements, ensuring hologram projection distances align with human eye appropriateness [58]. HoloLens utilizes an illumination sensor to adapt display brightness for ambient lighting, optimizing battery life. The camcorder captures the HoloLens environment and holograms in video format. For spatial mapping, HoloLens features its camera, creating a virtual 3D model of the user's surroundings [58, 59]. The HoloLens visor and screens restrict the viewing angle from 120° to 120° [58]. The 120° to 120° viewing angle not only dictates the hologram's position on HoloLens but also influences the recognition of hand movements for device control. This range encompasses the entire area where HoloLens holograms and functions are observable. HoloLens has built-in speakers just above the ears, facilitating the use of spatial sound [60]. Spatial sound enhances immersion by delivering varied sound distances and locations to the user. It serves as a tool to heighten the mixed-reality experience and provides distinctive opportunities for HoloLens use [58].

HoloLens incorporates a CPU, GPU, and sensors in an Inertial Measurement Unit (IMU) to gauge the environment and user movements. The IMU, using a gyroscope and accelerometers, tracks the HoloLens user's position and location [58, 61, 62].

HoloLens employs the Intel Atom × 5-Z8100 processor, featuring 14 nm technology and 4 logical cores. The graphics are handled by the HoloLens Graphics card. The device offers

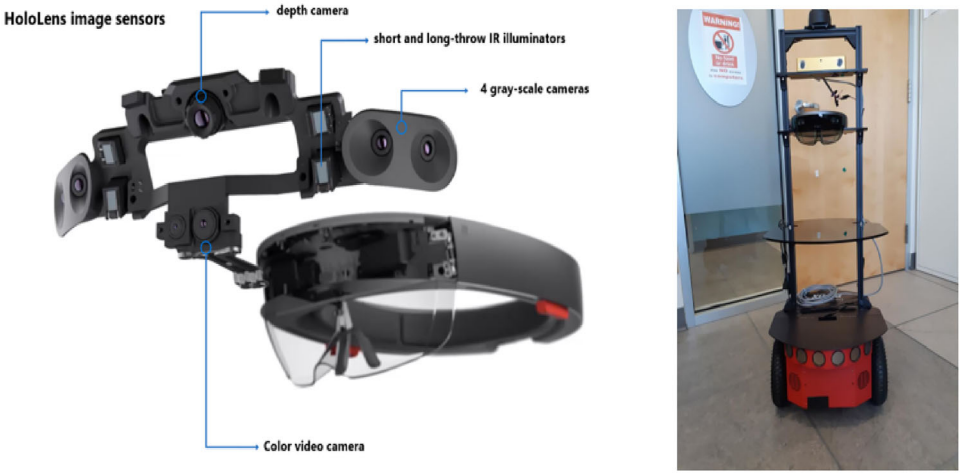


Fig. 7 Overview of Microsoft's HoloLens

64 GB of storage, with approximately 54.09 GB available for applications, videos, and other data [60].

HoloLens is used with voice commands or with pre-taught commands made by hand and head. The Head Position (Gaze) acts as a cursor or pointer to the Windows 10 operating system that is visible on HoloLens. At the same time, the Gaze function works for HoloLens as an identifier of what the user is currently experiencing through the displays [58]. Opening or selecting applications by combining the Gaze function, as well as a voice command or a business command (Gesture). When using a voice command, the microphone built into HoloLens recognizes the word "Select" as the dial tone. When using a business command, the right- or left-hand finger is lifted straight with the other fingers in the fist. To open an application or select, move the forefinger in a downward motion (Air tap). Combining Gaze and Gesture's Air Tap command creates the cursor movement and selection of the object. However, when using the Gesture function, you must remember the HoloLens restricted area 120° times 120°, which is why the function should be performed in the reserved area. HoloLens' pre-educated Gesture functions are also available in the Bloom and Tap and Hold commands. The purpose of the Bloom command is to open the "Star Menu" in the HoloLens Microsoft 10 operating system. The purpose of the Tap and Hold command is to navigate through pages and files for moving and zooming [58].

Note that obtaining raw IMU sensor data on HoloLens is not possible and will not be utilized in this project. In addition, in 2018, Microsoft introduced HoloLens Research Mode, offering application access to imaging sensor data on the device, including tracking cameras, depth camera data, and the IR-reflectivity stream [63].

Virtual landmark hologram app In this work, the virtual landmark hologram was produced for display on HoloLens for first and then real-time robot application using Unity3D [62], a cross-platform game engine with capabilities to create and edit 2/3D virtual models for HoloLens. The hologram made with Unity2/3D is then transferred wirelessly to HoloLens using Remote Control via a WiFi connection [64]. The use of this program is based on the simplicity of moving these holograms and moving even more complex holograms naturally. With Remote Control for HoloLens, one can transfer holograms directly from the Unity3/2D engine to HoloLens without additional build-in solutions through Visual Studio.

Figure 8 shows a standard Unity3D engine window of HoloSLAM virtual landmark hologram app. For HoloLens, the Unity3D gaming engine player and quality settings must be changed to HoloLens and other Microsoft products. The most important thing is to change the distance at which the hologram appears in the lenses. Changing the distance takes place by changing the Near Clip Plane distance from the Main Camera object settings of the Unity3D game engine under Hierarchy and Inspector. The recommended distance for HoloLens is 0.85 m. HoloLens is, therefore, recommended that the holograms are located at 1.25 m from the user [58]. Other considerations are in camera settings. From the Main Camera settings, the Clear Flags component must be changed from SkyBox to Solid Color. HoloLens makes the black background completely transparent, so the Solid Color RGBA values are changed to zero.

The quality of the Unity3D gaming engine is changed from Edit, Project setting and Quality, and then the menu opens. From the menu, select the item with the Windows operating system logo. For HoloLens, the quality settings are changed to the lowest level, so that the menu is set to Very Low for the best performance of HoloLens, and no delay occurs when looking at the hologram. The final step in the work is creating the hologram itself. This project chose three distinct virtual 3D landmarks, comprising a cube, diamond, and sphere, for the holographic representation. To create a

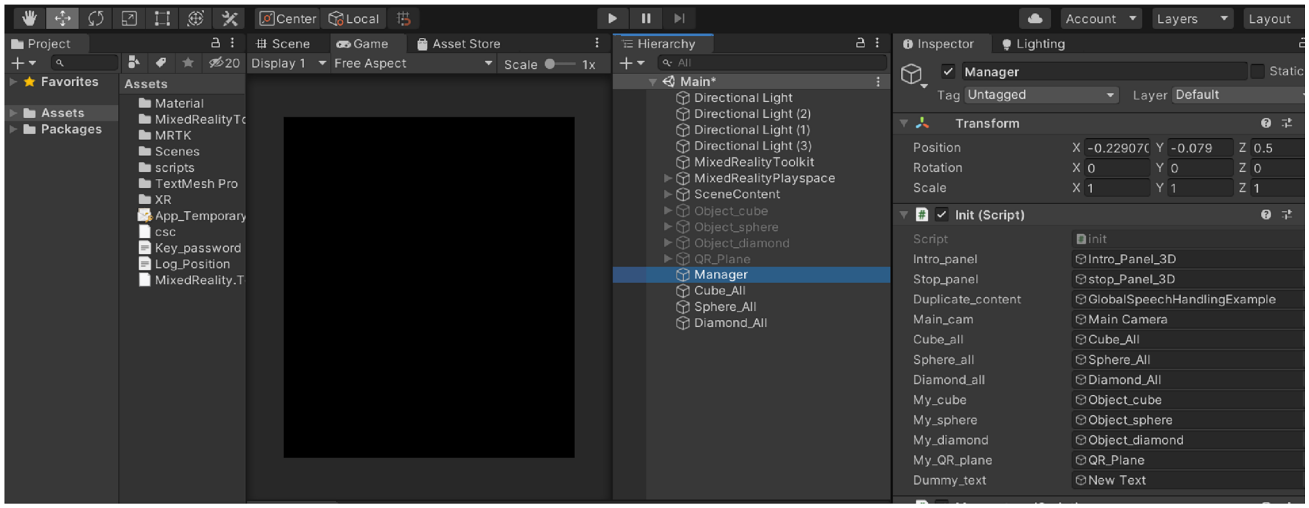


Fig. 8 Unity3D Window of HoloSLAM virtual landmark hologram app engine

cube for example, select Create from the Hierarchy panel. From the Create menu, a second menu opens, from which 3D object and Cube is selected. The cube will then appear in the Unity3D game engine preview window. The cube settings are further selected from Inspector, Transform, and then from Position to cube x, y and z coordinates of 0, 0, 1.25, so that the cube is about 1.25 m from the coordinate of the viewer. In the same menu you can find the Rotation whose values can be adjusted as desired, as well as the Scale, whose x, y and z values have been changed at work 0.25. Scaling has been reduced to make the cube created in HoloLens look appropriate. It is not possible to use the HoloLens emulator or the simulation of the correct HoloLens unless the Windows version of the computer you are using is compatible with Hyper Virtualization [65]. This feature is supported only by Windows Pro or Enterprise versions.

HoloLens seamlessly integrates virtual objects into the real world through spatial mapping and tracking technology. Utilizing HoloLens advanced sensors, cameras, and algorithms, it ensures that virtual objects maintain their position and perspective relative to the environment, adapting to changes in the user's viewpoint or position. The device employs sensors like depth cameras and inertial measurement units (IMUs) to create a detailed 3D spatial map of the environment, continuously updating it to accommodate changes [66]. This spatial mapping capability distinguishes HoloLens as a mixed-reality (MR) device, differentiating it from augmented reality (AR) devices.

Spatial mapping involves generating a 3D model of physical space using sensors, providing insights into the spatial layout and positioning of objects. Scene understanding interprets elements within the environment, recognizing object attributes. In Unity, 2D/3D spatial mapping utilizes depth data to create a 3D mesh, updating in real-time.

The triangular mesh in the spatial map links to a world-locked spatial coordinate system, ensuring consistency in virtual object placement. However, there is a trade-off between mesh density and processing power, requiring balance for a smooth MR experience. The mesh is continuously refreshed based on user and environmental movements.

Virtual objects are placed using spatial anchors, fixed points in physical space. HoloLens adjusts objects in real-time based on the user's movements, applying perspective corrections for a consistent appearance.

As of now, Microsoft has not released official documentation detailing the integration of algorithms and hardware within the HoloLens. The precision of the sensors and the placement of virtual objects remain undisclosed. Based on various cases and experiments [67], it has been observed that under certain conditions, the accuracy of position data can be achieved within $\pm$ a few centimeters. Therefore, in our practical experiments and during the implementation of HoloSLAM, we take this margin of error into account.

Proposed system and methodology

The Microsoft HoloLens mixed-reality technique and landmark-based SLAM approach (Ellipsoidal-SLAM) is used with a humanoid robot (Nao) to validate the implementation of HoloSLAM application in a real-world indoor environment application. To explain the details of the proposed algorithm and approach of HoloSLAM, we start this section by a short description of the Nao V5 humanoid robot and its technical and mechanical characteristics. We will mainly explain the sensors used in this work including Nao audio system, sonar, and Nao's odometry problem. This background information is required before introducing the final implementation of HoloSLAM.

Platform and software (Nao humanoid robot)

Nao, a medium-sized humanoid robot, developed and produced for a first time by a French company “Aldebaran “2004 by Bruno Maisonnier [68]. In 2008, the company launched the Nao Academics Edition, intended for education and university research. In 2014, Japanese company Softbank acquired this company and changed “Aldebaran “to "Softbank Robotics".

The main version of Nao is the H25, which has 25 degrees of freedom. Nao is 58 cm height and about 5 kg weight. The robot also has an Intel ATOM Z530 1.6 GHz processor, with 1 GB of RAM, as well as an Ethernet port, a WiFi connection, and a USB port. Figure 9 provides a summary of Nao robot and its main features including move, sense, communicate and think. Gouaillier [69] describes the details of the mechatronic design of Nao.

The main applications of Nao occurred in indoor building-scale environments, where navigation infrastructures like GPS is not available, hence the need for SLAM for Nao.

Since the first Nao version, the robot has two speakers located at the position where one would expect the ears of a human to be (see Fig. 10). These speakers are quite large compared to, e.g., smartphone speakers. However, their frequency range is specified to be from 0.2 to 10 kHz [70]. The older V3 version of the Nao has been designed to have a reasonably flat speaker response with a frequency range from 0.2 to 7 kHz [71].

The H25 version of the V5 Nao robot has four microphones that are positioned at the upper side of its head (see Fig. 11). They allow sampling rates up to 48 kHz when using a mix of all four channels and sampling rates up to 16 kHz when the individual channels are required [70]. The Nao’s microphones are significantly higher quality than the robot’s speakers, therefore, it is not expected that they will limit the transmission capabilities for acoustic communication [71].

Nao also is equipped with two ultrasonic sensors (or sonars) (two are transmitters and the other two are receivers) which allow it to estimate the distance to any obstacles in its environment. Ultrasonic sensor can detect an obstacle at a distance from 25 to 255 cm, but no distance information if the obstacle under 25 cm, the robot only knows that an object is present [70] (see Fig. 12).

Like other bipedal humanoid robots, Nao faces some specific challenges when it moves from one place to another on relatively flat surfaces. Nao rarely achieves its desired trajectory because of the deviation or bias generated by its odometry during the robot walking [26]. This problem is for different circumstances such as robot manufacturing errors, wear and tear of mechanic parts, or variations of floor flatness are because of partial hardware failures, out-of-specs components, the friction forces that are generated during motions, additional hardware mounted, friction forces generated during motion, etc. [72]. While walking, Nao needs to constantly adjust its pose estimate since high inaccuracies in its motion execution might lead to a deviation from the original motion plan. Therefore, it is also difficult for Nao to accurately follow a specified trajectory and corrects its motion without real/virtual landmark and using some probabilistic landmark-based filtering methods such as Kalman filter and its various nonlinear extensions such as extended Kalman filter (EKF) or Ellipsoidal-SLAM or other SLAM methods.

HoloSLAM algorithm implementation procedure

The Microsoft HoloLens mixed reality-SLAM (HoloSLAM) is used in this study to address most of SLAM landmark-related issues including landmark identification, data association, and reduce computational cost in a similar way to robotic augmented reality. With HoloSLAM, the robot can choose which virtual landmark to place in its environment and modify them in the robot environment in real-time when needed using its own voice. The robot then can interact with these virtual landmarks in a similar way to humans. The fundamental idea of using robot’s voice command is give the robot the ability to imitate humans when they perform mixed-reality technique.

HoloSLAM will provide a new robotic environment to the robotic SLAM literature. The approach is also optionally able to instruct the robot to navigate to a specific position by saying “go to the landmark” rather than giving an explicit navigation instruction.

HoloLens is mounted on Nao ’s head as shown in Fig. 13 and HoloLens’s camera is being used to capture QR codes which are used as artificial landmarks. The QR codes are strategically placed randomly in the robot operating environment to provide global pose references real landmarks for SLAM applications. These QR codes can be augmented with some information that can help to improve mapping process, reduce the computational cost and to solve data association in a way like RAR (robotic augmented reality).

Through our experiments, we have discovered two notable disadvantages of using QR codes as artificial landmarks for mapping the environment. First, detecting and extracting a bare QR code becomes quite challenging when it is beyond a distance of 1 m, which is a common condition for indoor mobile robots. Second, the quality of QR code detection is highly reliant on the angle between the camera and QR code planes. Only when these planes are almost parallel can acceptable quality and robustness be achieved. However, with the implementation of our HoloSLAM method, most of the issues related to landmarks and their detection can be resolved.

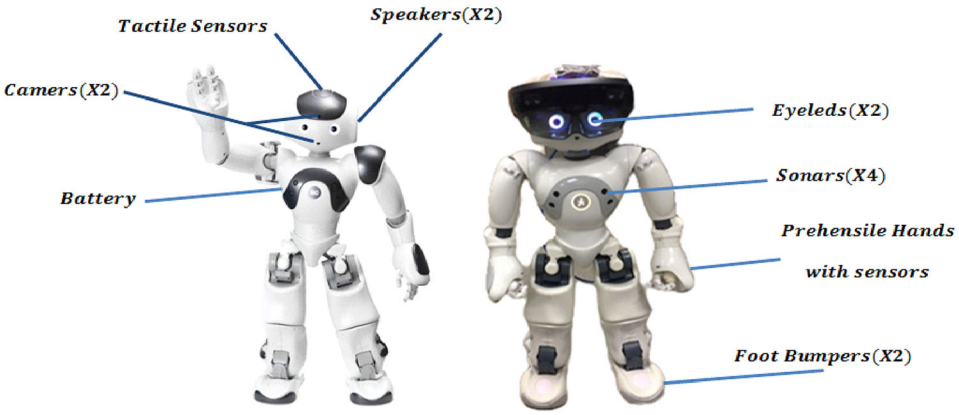


Fig. 9 Nao robot's sensors

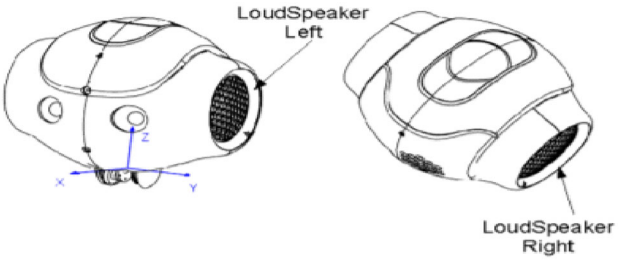


Fig. 10 Nao robot's main speakers' locations

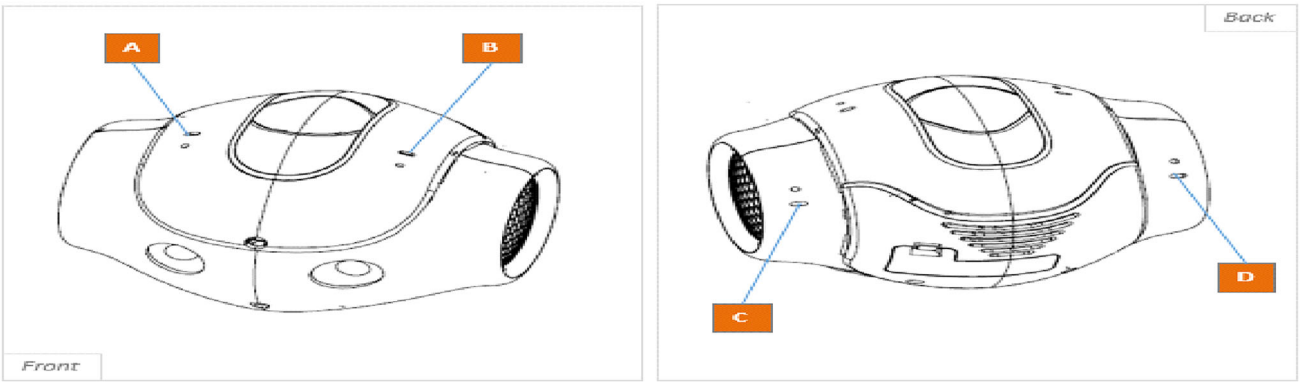
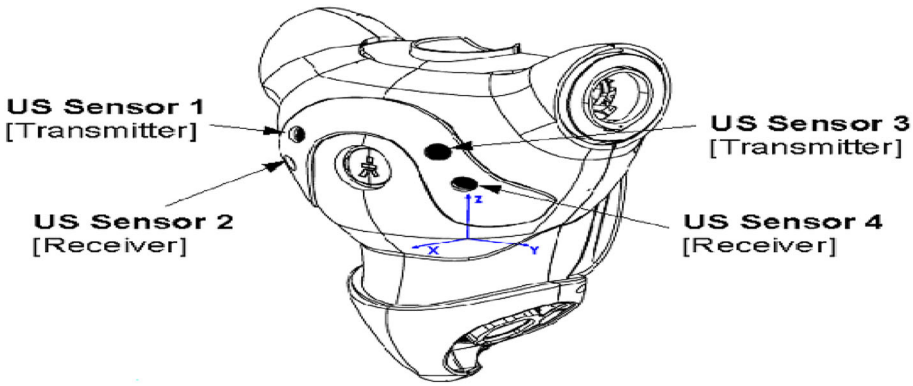


Fig. 11 Nao robot's main speakers' locations

Fig. 12 Nao robot's main sonars



To obtain the HoloLens's camera parameters in each distance with respect to QR codes, HoloLens's camera calibration needs to be done. We utilize the cameras of the HoloLens, which operate based on the Pinhole camera model, also referred to as the perspective camera model. This is a widely used and recognized method for detecting QR codes

Fig. 13 Nao robot acquires QR code data using HoloLens

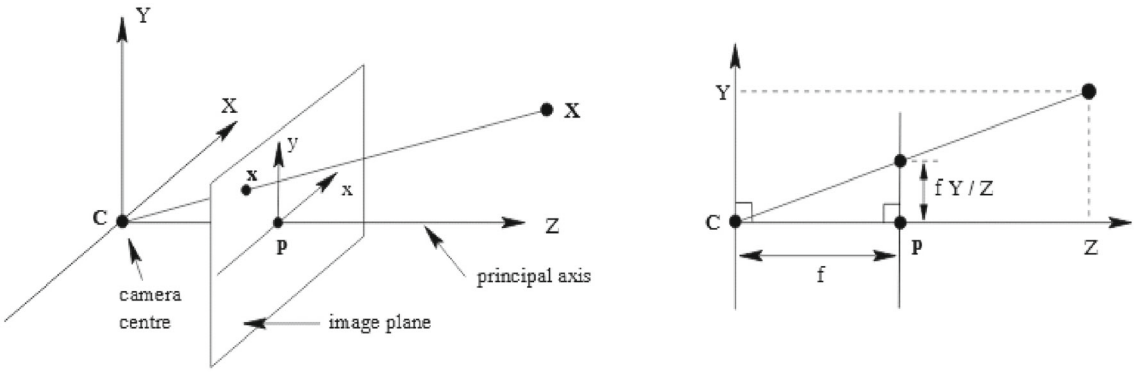


Fig. 14 Pinhole camera model. A Point pin 3D space is mapped to a 2D point p on image plane I by the ray connecting P with the center of projection C [75]

as realistic and natural landmarks in robotic environments. Pinhole model provides us with a mathematical relation between the points at 2D image plane and the re-projected 3D points in world coordinates [73, 74]. For calibration, at first, we need to understand the relationship between the four plane coordinate systems of camera model, namely the pixel plane coordinate system (**u, v**). The camera coordinate system ($\boldsymbol{X_C}$,$\boldsymbol{Y_C}$, $\boldsymbol{Z_C}$) and the world coordinate system ($\boldsymbol{X_w}$,$\boldsymbol{Y_w}$, $\boldsymbol{Z_w}$) are shown in Fig. 14.

The pinhole camera model is described as [76]

$$\boldsymbol{Z_c}\begin{pmatrix}\boldsymbol{u}\\ \boldsymbol{v}\\ 1\end{pmatrix} = \begin{bmatrix}1/\boldsymbol{d_x} & 0 & \boldsymbol{u_0}\\ 0 & 1/\boldsymbol{d_y} & \boldsymbol{v_0}\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\boldsymbol{f} & 0 & 0 & 0\\ 0 & \boldsymbol{f} & 0 & 0\\ 0 & 0 & 1 & 0\end{bmatrix}\begin{bmatrix}\boldsymbol{R} & \boldsymbol{t}\\ 0^r & 1\end{bmatrix} \quad (10)$$

$$= \begin{bmatrix}\boldsymbol{f_x} & 0 & \boldsymbol{u_0} & 0\\ 0 & \boldsymbol{f_y} & 0 & 0\\ 0 & 0 & 1 & \boldsymbol{v_0}\end{bmatrix}\begin{bmatrix}\boldsymbol{R} & \boldsymbol{t}\\ 0^r & 1\end{bmatrix}\begin{bmatrix}\boldsymbol{X_w}\\ \boldsymbol{Y_w}\\ \boldsymbol{Z_w}\\ 1\end{bmatrix} = \boldsymbol{M_1 M_2 X_w} = \boldsymbol{M X_w} \quad (11)$$

In this linear model, $\boldsymbol{f_x} = 1/\boldsymbol{d_x}$, $\boldsymbol{f_y} = 1/\boldsymbol{d_y}$ and which $\boldsymbol{d_x}$ and $\boldsymbol{d_y}$ are representing for the physical size of each pixel on the X axis and Y axis direction, and $\boldsymbol{f}$ is the focal length; $\boldsymbol{M_1}$ is a $3x4$ projection matrix; $\boldsymbol{M_1}$ known as the camera internal parameters is totally determined by $\boldsymbol{f_x}$, $\boldsymbol{f_y}$, $\boldsymbol{u_0}$ *and* $\boldsymbol{v_0}$, which are only related to the camera internal structure; $\boldsymbol{M_2}$ known as the camera external parameters is completely determined by the position of camera relative to the world coordinate system.

There are many different approaches to calculate the intrinsic and extrinsic parameters for a specific camera setup. In our approach, the camera parameters are obtained by a Canny Edge detection [77, 78]. The result is a transformation matrix which includes the (**x, y, z**) coordinates of the landmark in the robot frame.

Figure 15 illustrates the process of converting the global frame of the world to the local frames of the Nao robot using the HoloLens and QR codes landmark.

Typically, a 2D transformation that links two frames involves both rotation and translation. This type of transformation can be expressed as

$$\begin{bmatrix} \mathbf{X_{global}} \\ \mathbf{Y_{global}} \end{bmatrix} = \begin{bmatrix} \mathbf{cos}\varnothing & -\mathbf{sin}\varnothing \\ \mathbf{sin}\varnothing & \mathbf{cos}\varnothing \end{bmatrix} \begin{bmatrix} \mathbf{X_{QRcode/Hololens}} \\ \mathbf{Y_{QRcode/Hololens}} \end{bmatrix} + \begin{bmatrix} \mathbf{X}_0 \\ \mathbf{Y}_0 \end{bmatrix} \quad (12)$$

where "$\boldsymbol{QR/Hololens}$" indicates QR coordinates in the HoloLens/robot frame ($\boldsymbol{X}_{\boldsymbol{QR/Hololens}}$, $\boldsymbol{Y}_{\boldsymbol{QR/Hololens}}$) and $\boldsymbol{X}_0$ and $\boldsymbol{Y}_0$ are the robot location in the global frame. The angle $\varnothing$ denotes the orientation of the robot in the global frame. The equation can be rewritten as

$$\begin{bmatrix} \mathbf{X_{global}} \\ \mathbf{Y_{global}} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{cos}\varnothing & -\mathbf{sin}\varnothing & \mathbf{X}_0 \\ \mathbf{sin}\varnothing & \mathbf{cos}\varnothing & \mathbf{Y}_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{X_{QRcode/Hololens}} \\ \mathbf{Y_{QRcode/Hololens}} \\ 1 \end{bmatrix} \quad (13)$$

Now, it is possible to get the global coordinates of any QR code in the environment.

The pseudocode for the hologram app to capture QR code and places virtual landmarks using HoloLens mixed-reality technique is presented below. The execution of this hologram takes place within the HoloLens device, and it can only be initiated by a voice command *(Start)*issued through the Nao robot's speakers. The pseudocode is included below:

Start- *Launch the* ***Holo-landmark****hologram app*
voice function command (***Start*** *).*
Get Observation*– Capture a* QR code *using Hololens's camera and send it to Python code to localize it with respect to Hololens camera and then to robot body.*
Looking for QR codes***Yes -*** *turning* Nao*'s head by 30* degrees.
-*localize* QR codes *with respect to robot location.*
- voice function command ***(takepicture).***
No -*place virtual landmark.*
-voice function command ***(place virtual landmark).***
Exit- *close the* ***Holo-landmark*** *hologram app*
voice function (***Exit*** *)*

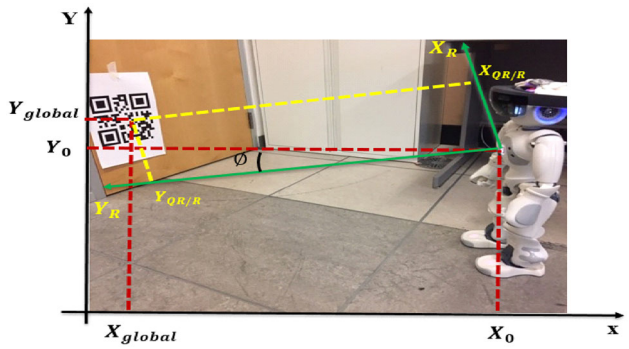


Fig. 15 QR code global coordinates

Initially, after deploying the *Holo-landmark* hologram app within the HoloLens device via the Nao robot's voice command function *(Start),* the robot can initialize the HoloLens and verify that the application is operational. The app has two main stages. In the first stage, the app involves the Nao robot and HoloLens to recognize some of QR codes and obtain some preloaded augmented information. In every head turn, Nao robot commands HoloLens to take picture and look for any QR codes inside. This is done by the voice command function *(takepicture)* in the Holo-landmark hologram app. Once the QR code is detected, any information augmented inside it will be captured and the QR code location will be localized with respect to robot frame and global frame respectively. This information can later be used for the simplification of data association problem in SLAM and help navigate in the robot environment. In the next stage, the robot commands HoloLens and the Holo-landmark hologram app using voice function command *(place virtual landmark)* to place virtual landmark in its environment. This helps the robot to perform SLAM even though no real landmark is detected. The *Holo-landmark* hologram app developed in this work is designed to allow the placement of a single landmark, whether it be a cube, sphere, or diamond shape, each time the voice function command *"place virtual landmark "*is initiated. However, it is possible to include additional objects or landmarks by simply adding new functions such as "*place sphere*", "*place desk*," and "*place virtual robot*". Unity3D has many ready 2/3D objects that can be easily used as a landmark. In addition, some special landmarks can be designed using 2/3D software design such as Morphi, 3D Slash, Fusion 360, and Blender [79]and then can be imported

to Unity3D engine. The designed virtual landmarks can be randomly placed in any position. In our approach, the Nao robot's sonar is utilized to determine a secure distance for the robot to navigate and avoid any obstructions while executing the SLAM algorithm. Within this distance, the robot can place a virtual landmark.

Other functions can be added to the *Holo-landmark* hologram app to give the robot the ability to interact with landmark such as move*(up),* move*(down),* and move*(right).* All these functions are tested in our main hologram. The added function depends on the robot application and app functionalities. *Finally, the voice function (Exit*) is used to give the robot the ability to stop the app and exit for any reason. The pseudocode for the HoloLens mixed reality-SLAM integrated with EKF/Ellipsoidal-SLAM of is presented below:

Start-
Initialization*- SLAM Initialization,* Nao *Robot Initialization,* ***Launch Holo-landmark*** **hologram app (*voice function command(start)*).**
Get Observation –
Looking for QR codes*(turning*Nao*'s head by 30* degrees) *(**voice function command (takepicture)).***
Yes-localize the QR codes with respect to robot location.
No - Hologram-landmark app launched.
- Place Virtual Landmark (***voice function command (place virtual landmark))***
while not_stop
Prediction Step *- Check safe distance to move by sonar. (**Move command**)*
No*- safe distance. Turn 180 degree*
Get Observation (4)*- Looking for* QR codes *by turning head by 30* degrees.
If Nao *find QRcodes to robot frame*
Yes *– Are there any Augmented* QR codes.
No - *Use virtual Landmark Holograms*
launch voice function command ***(place virtual landmark).***
Data Association-
Real landmark and Virtual Landmark matching and data- association simplification
Correction _Step - *Run* standard *EKF/ Ellipsoidal - SLAM update step.*
Augmented _Map- *Add new Real and Virtual Landmarks to the map.*
Check if iteration numbers are achieved.
No *Go to step 4*
End- Close-Holo-landmark **hologram app.** ***Voice function command (Stop)***

The EKF/Ellipsoidal-SLAMs perform regular map initialization process and at the same time, the robot starts the Holo-landmark hologram app using his voice command (start). Nao starts turning its head by *30*-degree steps and at the same time looking for QR codes. In each head turn, the robot commands the HoloLens to take picture and look for QR codes inside it and once any of QR code is detected, it will get the augmented data and all information associated with it and at the same time it calculates the QR code coordinates with respect to the robot frame. If the robot did not find any QR code, it will place virtual landmark using voice function command *(place virtual landmark).* The sonar checks available space before the robot moves to the for next prediction step.

The fundamental structures of EKF-SLAM and Ellipsoidal-SLAM remain while the HoloLens mixed-reality processes take place when virtual landmark is placed and where additional information is retrieved regarding to the detected landmark in both virtual and real landmark. ALL detected virtual/real landmarks are then mapped to global location using EKF/Ellipsoidal-SLAM.

The implemented SLAM (HoloSLAM) is designed to work using only virtual landmarks to map the environment or using a mixed environment that includes real and virtual landmarks at the same time. Our goal is to map the robot environment using both kinds of landmarks.

The main contribution of integrating HoloLens mixed reality into EKF-SLAM or Ellipsoidal-SLAM is the use of the virtual landmarks when there are not any real landmarks in the environment to avoid any failure to SLAM process and to use the augmented additional information in virtual and real landmarks. This will improve the performance of EKF/Ellipsoidal-SLAM in terms of reducing the computational effort, simplifying the data association problem, improving the SLAM algorithm, and assisting robots in navigation tasks within a practical environment.

Experimental results

HoloSLAM was assessed in the Autonomous and Intelligent Systems Laboratory (AISL), where QR codes were positioned at undisclosed spots throughout the lab, aligned with the Nao robot's height as shown in Fig. 16. The robot rises and commences its quest for landmarks (QR codes) by rotating its head 30°.

The objective is to enable the Nao robot to move independently while continually estimating its position and creating a map of the surrounding environment. The robot initiates movement and utilizes its sonar sensor to ensure a safe distance. With each step, the robot is instructed to proceed straight using a move command, and at each instance of motion, the Nao employs the HoloLens device affixed to its head to search for landmarks (QR codes).

Two distinct scenarios are under investigation. The first scenario entails examining the robot's operation in an environment devoid of any artificial landmarks (QR codes) and the second scenario involves a combination of virtual and artificial landmarks.

First scenarios: virtual HoloSLAM (SLAM with virtual landmark only). In this scenario, the Nao robot was placed in an environment free of any QR codes, as shown in Fig. 17. The experiment will assess how effectively the robot can estimate its position in the absence of physical or artificial landmarks. By relying solely on virtual landmarks, the HoloSLAM algorithm will be evaluated for its ability to provide accurate position estimates. The robot is tasked with placing virtual objects in its environment without any prior knowledge or human assistance. The aim is to evaluate the effectiveness of utilizing these virtual landmarks exclusively in the HoloSLAM algorithm, which facilitates the robot's navigation task. The robot builds a virtual map of the environment and can experience mixed reality in a similar way to the human. The robot places virtual objects in its environment without prior knowledge and without any human help. The Nao's ability to estimate its position and construct an accurate map of its environment with the absence of real or artificial landmarks would provide valuable insights into the usefulness of virtual landmarks in the SLAM algorithm.

The robot is expected to move in a straight line due to the given command; however, it deviated to the left and right because of the uncertainty in the odometry systems. The lack of landmarks leads to an increase in uncertainty error in every movement, resulting in eventual inaccurate SLAM. The measurements from feet encoders are subject to inherent errors and uncertainties that accumulate over time. As a result, the predicted robot position based solely on odometry tends to deviate from the actual path the robot has traveled. This error could be minimized using external sensor data to correct the robot's position using the Ellipsoidal-SLAM algorithm. As a result of this error, the odometry alone cannot be relied upon for accurate navigation tasks. Figure 18 shows the uncertainty of the estimated robot motion, which is represented by an ellipse, along with robot's real trajectory.

At each step, the Nao robot systematically examines its surroundings to detect QR codes. In the absence of QR codes,

Fig. 16 Experimental environment

Fig. 17 Virtual HoloSLAM experimental environment

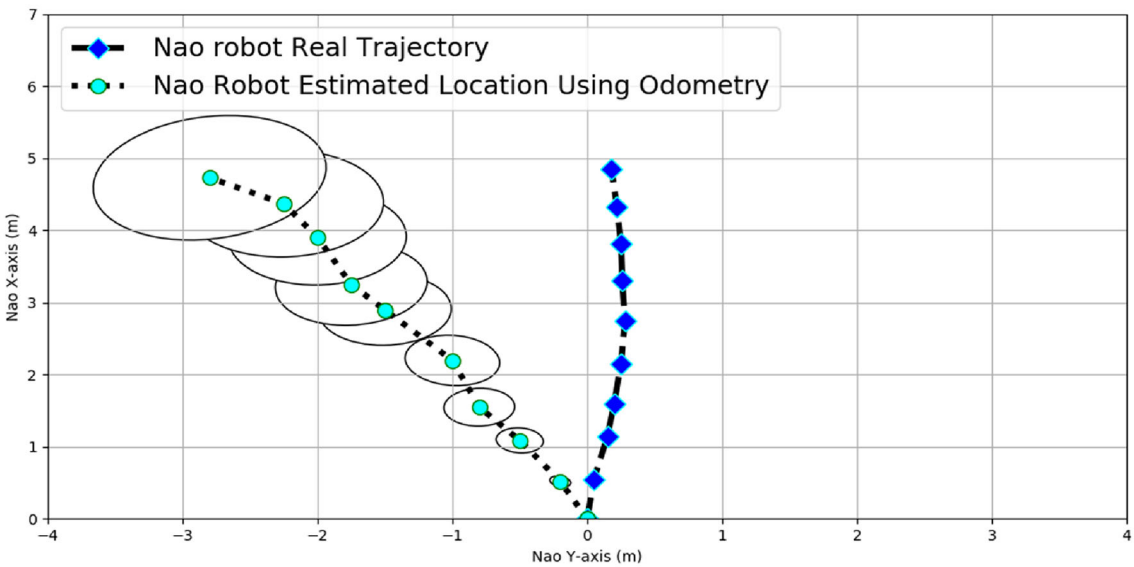


Fig. 18 Estimated and real robot position

the robot can autonomously place one or more virtual landmarks positioned at predetermined distances. Our virtual landmark app provides a selection of three distinct virtual landmark types, including diamonds, spheres, and cubes.

In this experimental setup, the robot receives instructions to position a virtual landmark in space, leaving it in place, and subsequently capture images, as illustrated in Fig. 19. The virtual objects are consistently positioned 2 m along the x-axis of the Nao robot. The orientation of the robot's body determines the angle at which these virtual landmarks are situated.

With HoloSLAM, the robot gains the capacity to precisely position virtual landmarks, eliminate them when they become unnecessary, and establish real-time communication with them. In addition, it empowers the robot with greater autonomy and control over its build map. Consequently, this ensures that the SLAM algorithm remains reliable even if the robot's sensors fail to detect any of the landmarks during the observation step.

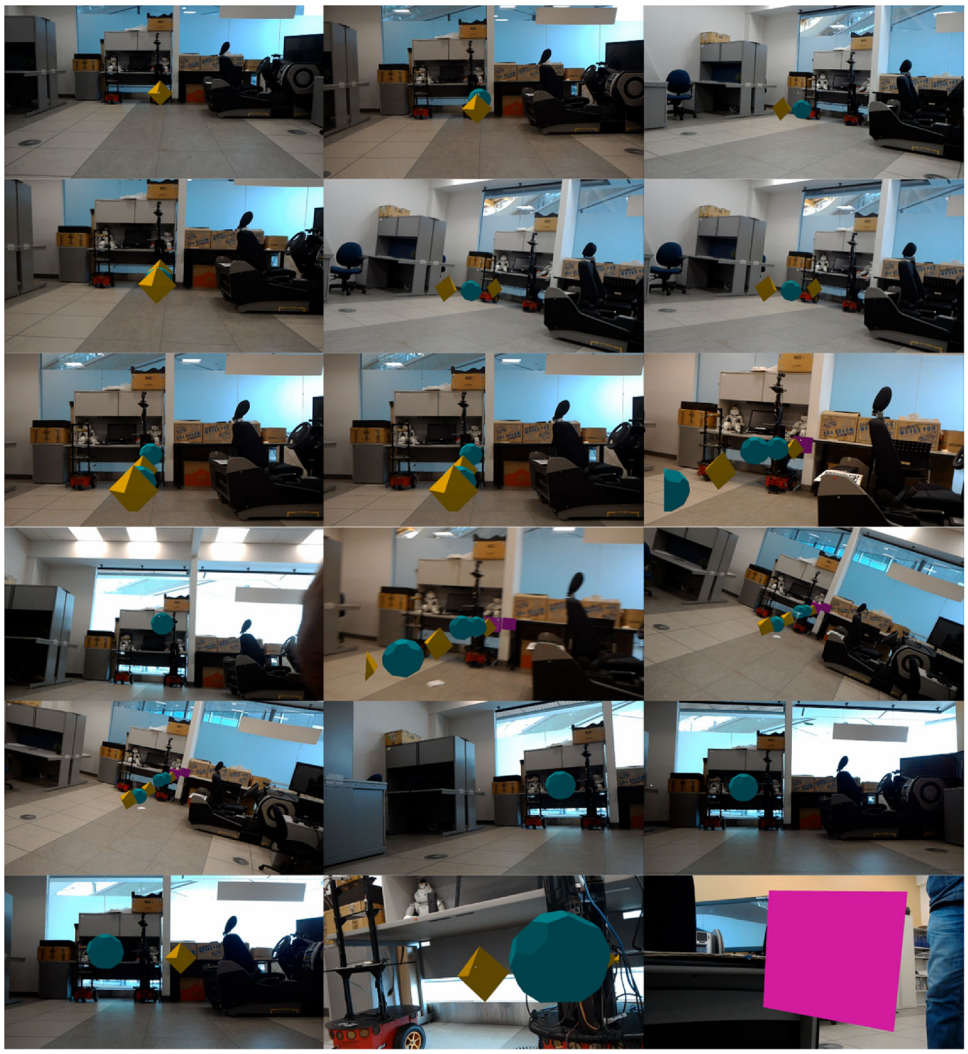


Fig. 19 Various virtual landmarks placed by the robot

The new virtual SLAM state vector now has only information about the robot's location and virtual object in its environment with its locations as follows:

$$\mathbf{s_t} = (\mathrm{p_{x,t}}, \mathrm{p_{y,t}}, \mathrm{p_{\theta,t}})\mathbf{Cube}\big(\mathrm{vm_{x,1}}, \mathrm{vm_{y,1}}\big), \ \mathbf{Sphere}(\mathrm{vm_{x,2}}, \mathrm{vm_{y,2}}) \ldots\ldots, \mathbf{Diamond}(\mathrm{vm_{x,N}}, \mathrm{vm_{y,N}})^{\mathrm{T}}$$

where ***vm*** indicates the virtual landmark type.

In Fig. 20, the virtual Ellipsoidal-HoloSLAM depicts the estimated robot position. Remarkably, the estimated trajectory closely aligns with the ground truth of the robot's position, as illustrated in Fig. 18. This suggests that the SLAM algorithm employed, Virtual Ellipsoidal-HoloSLAM, is performing well in accurately estimating the robot's position. Figures 18, 19, 20 illustrate that the estimated robot trajectory with virtual Ellipsoidal-HoloSLAM closely aligns with the actual robot position. This is attributed to the HoloLens, which furnishes a highly accurate position for the virtual landmark. This, in turn, significantly diminishes the error in Nao odometry.

Figures 21, 22 depict the actual and estimated positions of virtual landmarks using the virtual HoloSLAM system. *Second scenarios: HoloSLAM (combination of virtual and artificial landmarks).* This scenario involves placing QR code landmarks in the path of the robot, as shown in Fig. 23. Some of which are positioned in locations that are challenging for the robot to scan. The objective is to achieve precise SLAM and navigation when landmarks are scarce or hard to detect due to either issues with the detection system or the computational cost involved in image processing. In addition, the robot can place some virtual landmarks like the first scenario and interact with them. The robot can hide, remove, move up, move down, etc. of this virtual landmark. The robot is commanded to place the virtual landmark with a random angle.

Based on real-time experiments, we have observed certain challenges associated with QR codes. First, it is crucial to adjust the image resolution based on the distance between the

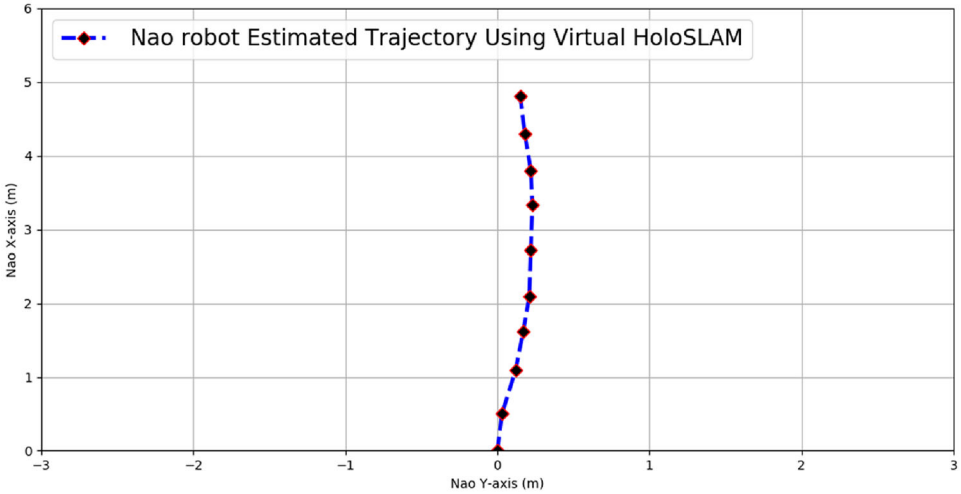


Fig. 20 Estimated robot position using virtual HoloSLAM

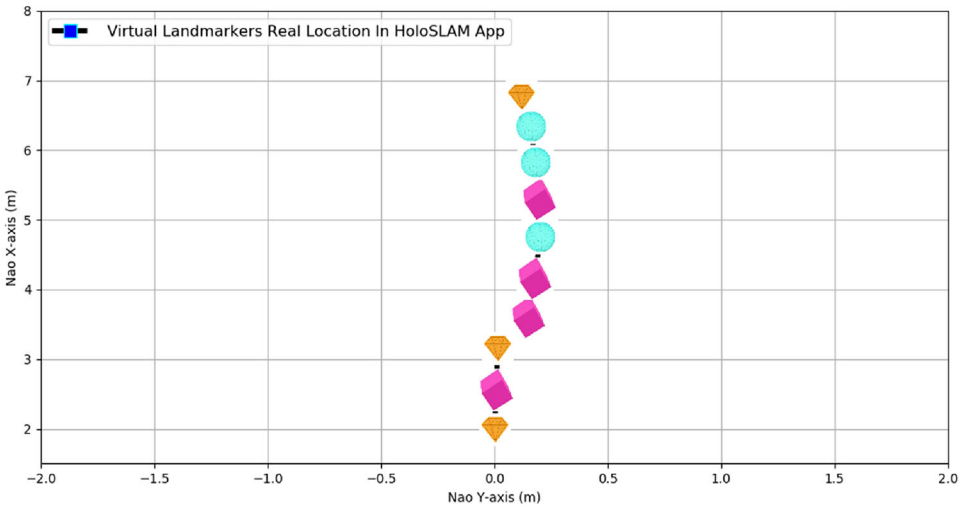


Fig. 21 Estimated real virtual landmarks position using virtual HoloSLAM

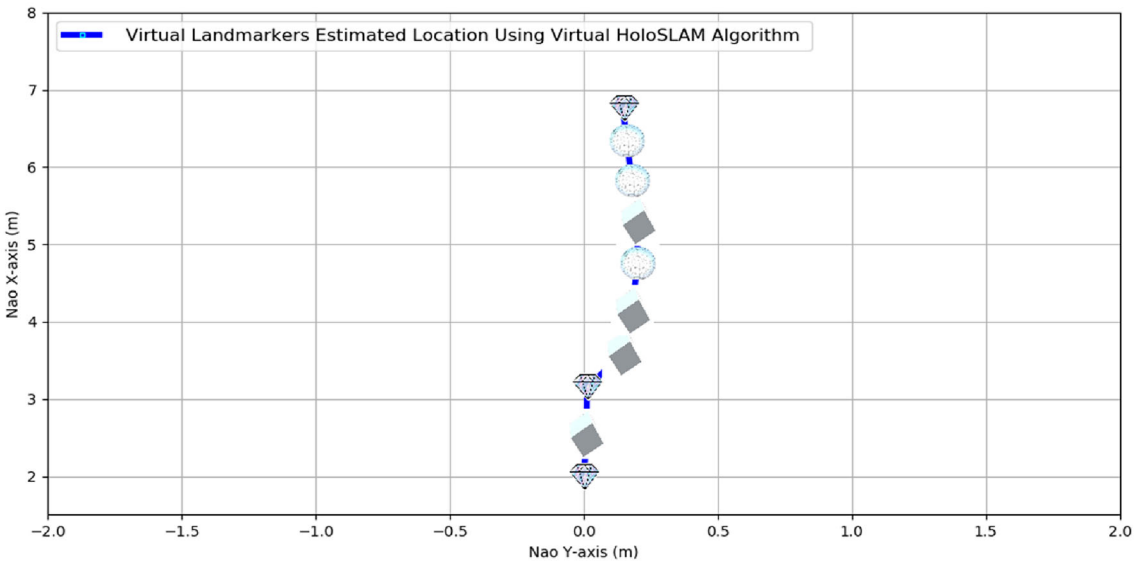


Fig. 22 Real virtual landmarks position using virtual HoloSLAM

Fig. 23 HoloSLAM experimental environment with QR codes

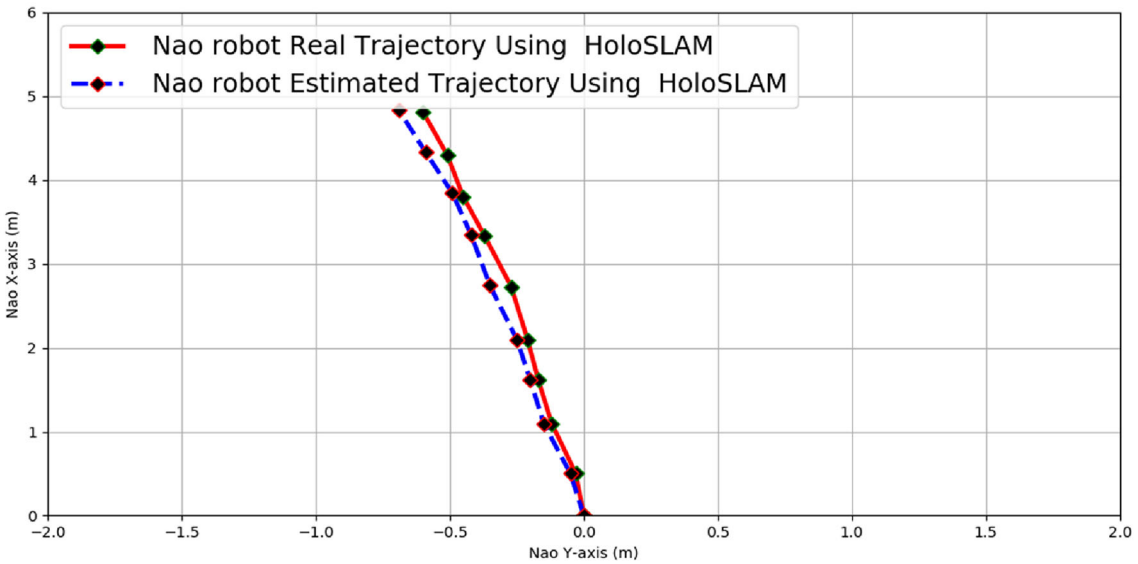


Fig. 24 Robot position using HoloSLAM algorithm

camera and the code. Second, the Nao robot can detect the designed QR codes effectively within a maximum distance of approximately 2 m. However, this range diminishes when the code is not directly within the camera's line of sight. The distance at which a QR code can be successfully scanned depends on factors such as the resolution and quality of the scanning device, lighting conditions, QR code size, and any potential obstructions between the QR code and the scanner. It is important to note that while increasing the size of a QR code can improve its readability at a distance, there are practical limits to consider. Extremely large QR code may not be suitable for certain applications due to space constraints or the need for high-resolution printing.

The position accuracy of QR code is less accurate than when using virtual HoloSLAM only. The odometry error and QR code calculated position error accumulated oversteps. The new SLAM state vector now has mixed information for both virtual and real landmarks. The object in its environment with its locations as follows:

$$\mathbf{s_t} = \left(\mathrm{p_{x,t}}, \mathrm{p_{y,t}}, \mathrm{p_{\theta,t}}, \mathbf{QRcode1}\left(\mathrm{vm_{x,1}}, \mathrm{vm_{y,1}}\right), \mathbf{Sphere}\left(\mathrm{vm_{x,2}}, \mathrm{vm_{y,2}}\right)\ldots, \mathbf{QRcode2}\left(\mathrm{vm_{x,2}}, \mathrm{vm_{y,2}}\right)\ldots, \mathbf{Diamond}\left(\mathrm{vm_{x,N}}, \mathrm{vm_{y,N}}\right)\right)^{\mathrm{T}}$$

QR codes are augmented with some information to help simplify the data association and reduce the computational cost. The estimated, real robot trajectory, QR code, and virtual landmarks after 10 iterations are shown in Figs. 24, 25, 26.

The RMS is calculated as follows: the evaluation of the implemented HoloSLAM included an analysis of the accuracy and consistency of both the state vector and the estimated

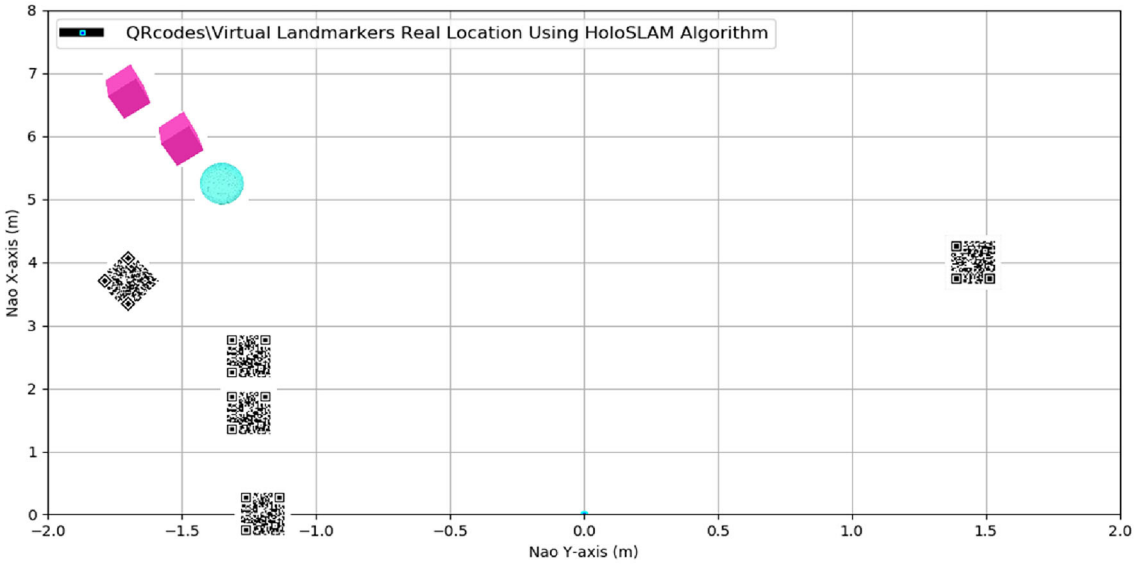


Fig. 25 Real and virtual landmark real positions using HoloSLAM algorithm

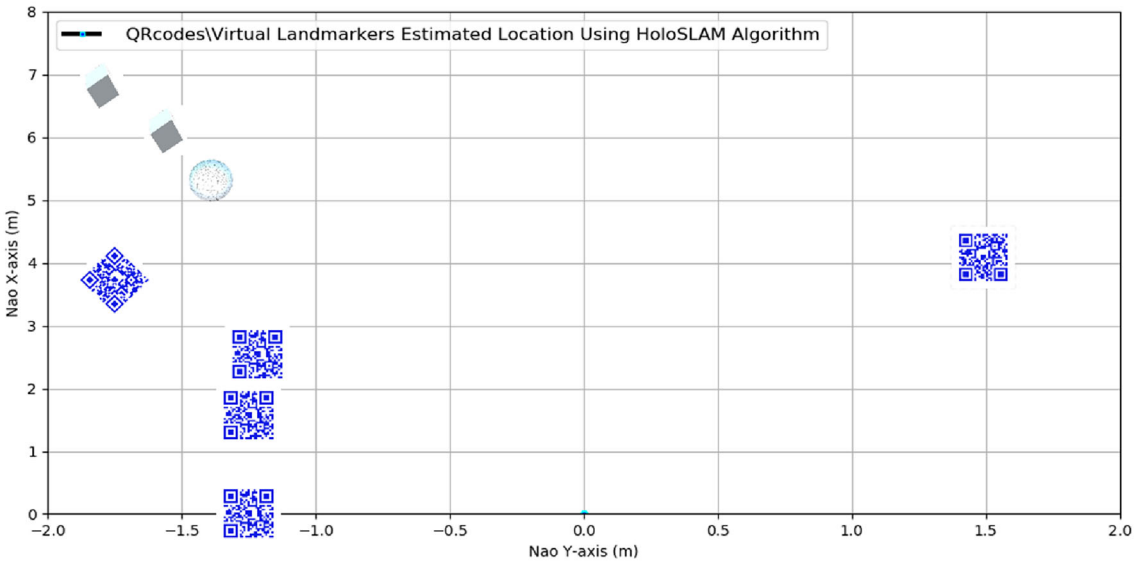


Fig. 26 Real and virtual landmark estimated positions using HoloSLAM algorithm

Table 1 Analyzing the effectiveness and performance of implemented SLAM algorithms

Algorithm	Nao position error/m	Nao orientation error/rad	Real landmark error/m	Virtual landmarks error/m
Nao IMU	50.01	0.78558	–	–
Ellipsoidal-HoloSLAM	0.0184	0.17975	0.1413	0.4022
Virtual Ellipsoidal-HoloSLAM	0.09010	0.10128	–	0.2044

positions of landmarks. We utilized the root mean square (**RMS**) method to measure these metrics. The RMS is defined as

$$\mathbf{e} = \sqrt{\frac{\sum_{\mathbf{i}=1}^{\mathbf{n_x}} (\mathbf{x_i} - \widehat{\mathbf{x}}_\mathbf{i})^2}{\mathbf{n_x}}} \tag{14}$$

The RMS is computed by taking the square root of the average of the squared differences between the true values ($\boldsymbol{x_i}$) and the estimated values ($\widehat{\boldsymbol{x}}_{\boldsymbol{i}}$) of the state vector. Similarly, the estimated positions of landmarks were also assessed using the RMS method.

Table 1 presents detailed results obtained from real-time experiments, where the values listed are the average RMS values across these experiments. These results provide insights into the overall performance of the HoloSLAM algorithm in terms of accuracy and consistency. Furthermore, the estimated positions of the virtual landmarks were also evaluated. As the table shows, the Nao's IMU has the largest error in the robot position and orientation. Ellipsoidal-HoloSLAM reduces the heading error by 66.6% of the total error of the IMU, but this is not enough to perform an accurate SLAM.

The experimental findings not only affirm the superior performance of the virtual Ellipsoidal-HoloSLAM algorithms but also highlight their outperformance over the conventional Ellipsoidal-SLAM method. This heightened effectiveness can be directly attributed to the integration of Microsoft HoloLens, which furnishes exceptionally accurate positions for virtual landmarks within the HoloSLAM application. Consequently, HoloSLAM systems emerge as potent tools capable of substantially mitigating errors, thereby providing more precise estimates of the robot's pose and the locations of real QR codes.

Remarkably, both the virtual and HoloSLAM Ellipsoidal-SLAM approaches showcase exceptional performance in accurately determining the robot's position at each time step, displaying minimal errors when compared to conventional odometry or regular Ellipsoidal-SLAM methods. This heightened level of accuracy and reliability positions HoloSLAM as a valuable and effective solution for tasks requiring precise localization and mapping capabilities.

Conclusions

The primary goal of the project was to address the challenges associated with landmark-based SLAM systems and develop innovative solutions to overcome issues such as absences, scarcity, shortage, and inaccurate detection of landmarks in the environment. The implemented system utilizes mixed reality to provide virtual landmarks to model the environment and provide accurate solutions to SLAM problem and implement them on the Nao humanoid robot. Virtual HoloSLAM solutions, which integrate Microsoft HoloLens mixed-reality techniques with Ellipsoidal-SLAM were explained and tested on the Nao humanoid robot to enable the Nao to move through its environment. This groundbreaking SLAM algorithm empowers the robot with the unique capability to interact with and exert control over its environment. The fundamentals of each step of mapping and localization have been explained and implemented with the Nao robot for Ellipsoidal-SLAM algorithms. The results of the experiments performed in the AISL lab showed that regular virtual Ellipsoidal-HoloSLAM algorithm has a better performance than traditional Ellipsoidal-SLAM in terms of mapping the environment. Virtual Ellipsoidal-HoloSLAM was more robust in modeling the motion errors. Besides the improvement of the algorithm itself to include some landmark information, we have reported great improvement in consistency and accuracy. Noticeably, virtual Ellipsoidal-HoloSLAM improves localization and mapping. The virtual Ellipsoidal-HoloSLAM was able to estimate the robot's position each time with very small errors when compared to localization done by odometry or by both HoloSLAM and regular Ellipsoidal-SLAM. The virtual Ellipsoidal-HoloSLAM results decreased IMU errors by 95%, while Ellipsoidal-HoloSLAM reduced this in the range of 70–80%. The experiments show that the robot can find its location and build an acceptable map around itself successfully at each step in an acceptable time using virtual Ellipsoidal-HoloSLAM.

Data availability The data is available upon request.

References

1. Jiang G, Yin L, Jin S, Tian C, Ma X, Ou Y (2019) A simultaneous localization and mapping (SLAM) framework for 2.5D map building based on low-cost LiDAR and vision fusion. Appl Sci. https://doi.org/10.3390/app9102105
2. Aulinas J, Petillot Y, Salvi J, Lladó X (2008) The SLAM problem: a survey. Front Artif Intell Appl. https://doi.org/10.3233/978-1-58603-925-7-363
3. Ding H, Zhang B, Zhou J, Yan Y, Tian G, Gu B (2022) Recent developments and applications of simultaneous localization and mapping in agriculture. J F Robot. https://doi.org/10.1002/rob.22077

4. Chen Y et al (2018) The accuracy comparison of three simultaneous localization and mapping (SLAM)-based indoor mapping technologies. Sensors (Switzerland). https://doi.org/10.3390/s18103228
5. Willners JS et al (2021) Robust underwater SLAM using autonomous relocalisation. IFAC-PapersOnLine. https://doi.org/10.1016/j.ifacol.2021.10.104
6. Kalita H, Gholap AS, Thangavelautham J (2020) Dynamics and control of a hopping robot for extreme environment exploration on the Moon and Mars. IEEE Aerosp Conf Proc. https://doi.org/10.1109/AERO47225.2020.9172617
7. Taheri H, Xia ZC (2021) SLAM; definition and evolution. Eng Appl Artif Intell 97:104032. https://doi.org/10.1016/j.engappai.2020.104032
8. Zhong X, Zhou Y, Liu H (2017) Design and recognition of artificial landmarks for reliable indoor self-localization of mobile robots. Int J Adv Robot Syst. https://doi.org/10.1177/1729881417693489
9. Nguyen X-H, Nguyen V-H, Ngo T-T (2020) A new landmark detection approach for slam algorithm applied in mobile robot. J Sci Technol - Tech Univ. https://doi.org/10.51316/30.7.6
10. Kumar Aggarwal A (2015) Machine vision based self-position estimation of mobile robots. Int J Electron Commun Eng Technol 6(10):20–29
11. Domain F, Jouffre D, Caverivière A (2000) SLAM for dummies: a tutorial approach to simultaneous localization and mapping. https://doi.org/10.1017/S0025315400002526
12. Básaca-Preciado LC et al (2020) Autonomous mobile vehicle system overview for wheeled ground applications. Machine vision and navigation. Springer, Cham. https://doi.org/10.1007/978-3-030-22587-2_15
13. Fazli S, Kleeman L (2007) Simultaneous landmark classification, localization and map building for an advanced sonar ring. Robotica. https://doi.org/10.1017/S0263574706003079
14. Souto LAV, Castro A, Gonçalves LMG, Nascimento TP (2017) Stairs and doors recognition as natural landmarks based on clouds of 3D edge-points from RGB-D sensors for mobile robot localization. Sensors (Switzerland). https://doi.org/10.3390/s17081824
15. Rosa S (2014) Localization and mapping for service robotics applications. p 113. https://doi.org/10.6092/polito/porto/2542488
16. Bahraini MS, Rad AB, Bozorg M (2019) SLAM in dynamic environments: a deep learning approach for moving object tracking using ML-RANSAC algorithm. Sensors (Switzerland). https://doi.org/10.3390/s19173699
17. Ryu H (2019) A revisiting method using a covariance traveling salesman problem algorithm for landmark-based simultaneous localization and mapping. Sensors (Switzerland) 19(22):4910. https://doi.org/10.3390/s19224910
18. Núñez P, Vázquez-Martín R, del Toro JC, Bandera A, Sandoval F (2008) Natural landmark extraction for mobile robot navigation based on an adaptive curvature estimation. Rob Auton Syst. https://doi.org/10.1016/j.robot.2007.07.005
19. Arican Z (2004) Vision-based robot localization using artificial and natural landmarks, August, 2004
20. Franco Jr JC (2019) Modelagem BIM de infraestrutura urbana a partir de levantamentos aéreos com drone
21. Holmes C, Barfoot TD (2023) An efficient global optimality certificate for landmark-based SLAM. IEEE Robot Autom Lett. https://doi.org/10.1109/LRA.2023.3238173
22. Gonzalez JP, Stentz A (2007) Planning with uncertainty in position using high-resolution maps. In: Proc. - IEEE Int. Conf. Robot. Autom. pp 1015–1022. https://doi.org/10.1109/ROBOT.2007.363118
23. Du ZJ, Huang SS, Mu TJ, Zhao Q, Martin RR, Xu K (2022) Accurate dynamic SLAM using CRF-based long-term consistency. IEEE Trans Vis Comput Graph. https://doi.org/10.1109/TVCG.2020.3028218
24. Dong X, Cheng L, Peng H, Li T (2022) FSD-SLAM: a fast semi-direct SLAM algorithm. Complex Intell Syst. https://doi.org/10.1007/s40747-021-00323-y
25. Atanasov N, Bowman SL, Daniilidis K, Pappas GJ (2018) A unifying view of geometry, semantics, and data association in SLAM. IJCAI Int Jt Conf Artif Intell. https://doi.org/10.24963/ijcai.2018/722
26. Lahemer ES, Rad A (2019) An adaptive augmented vision-based ellipsoidal slam for indoor environments. Sensors (Switzerland). https://doi.org/10.3390/s19122795
27. Aycock T (2010) A simultaneous localization and mapping implementation using inexpensive hardware. Update
28. Pedrycz W (1997) Bounding approaches to system identification. Control Eng Pract. https://doi.org/10.1016/S0967-0661(97)87398-4
29. Arena F, Collotta M, Pau G, Termine F (2022) An overview of augmented reality. Computers. https://doi.org/10.3390/computers11020028
30. Bray B, Zeller M, Schonning N (2018) What is mixed reality? Microsoft [Online]. Available: https://docs.microsoft.com/en-us/windows/mixed-reality/mixed-reality
31. Microsoft, "Microsoft, Microsoft HoloLens, Available https://www.microsoft.com/en-us/hololens", [Online]. Available: www.microsoft.com/en-us/hololens
32. C. Republic, A. Technologies and I. Studies (2019) Hybrid slam modelling of autonomous vehicle with, no. September
33. Fankhauser P, Bloesch M, Hutter M (2018) Probabilistic terrain mapping for mobile robots with uncertain localization. IEEE Robot Autom Lett. https://doi.org/10.1109/LRA.2018.2849506
34. Durrant-Whyte H, Rye D, Nebot E (1996) Localization of autonomous guided vehicles. Robot Res. https://doi.org/10.1007/978-1-4471-1021-7_69
35. Kalman RE (1960) A new approach to linear filtering and prediction problems. J Basic Eng. https://doi.org/10.1115/1.3662552
36. Samsuri SB, Zamzuri H, Abdul Rahman MA, Mazlan SA, Abd Rahman AH (2015) Computational cost analysis of extended Kalman filter in simultaneous localization and mapping (EKF-SLAM) problem for autonomous vehicle. ARPN J Eng Appl Sci 10(17):7764–7768
37. Sola J (2013) Simulataneous localization and mapping with the extended Kalman filter, Unpubl. Available http://www.joansola.eu/JoanSola/eng/JoanSola.html
38. Thrun S, Wolfram B, Dieter F (2005) Probabilistic robotics
39. Placed JA et al (2023) A survey on active simultaneous localization and mapping: state of the art and new frontiers. IEEE Trans Robot. https://doi.org/10.1109/tro.2023.3248510
40. Taheri H, Xia ZC (2021) SLAM; definition and evolution. Eng Appl Artif Intell. https://doi.org/10.1016/j.engappai.2020.104032
41. Cadena C, Carlone L, Carrillo H, Latif Y, Scaramuzza D, Neira J, Reid I, Leonard JJ (2016) Past, present, and future of simultaneous localization and mapping: towards the robust-perception age. IEEE Trans Robot. https://doi.org/10.1109/TRO.2016.2624754
42. Scholte E, Campbell ME (2003) A nonlinear set-membership filter for on-line applications. Int J Robust Nonlinear Control. https://doi.org/10.1002/rnc.856
43. Eliazar A, Parr R (2003) DP-SLAM: Fast, robust simultaneous localization and mapping without predetermined landmarks. In: IJCAI International Joint Conference on Artificial Intelligence
44. Werede Gunaza Teame, Yu Y, Zhongmin W (2020) Optimization of SLAM Gmapping based on Simulation. Int J Eng Res. https://doi.org/10.17577/ijertv9is040107
45. Abbeel P (2006) gMapping. Trans Robot
46. Haykin S (2001) Kalman filtering and neural networks. Wiley. https://doi.org/10.1002/0471221546
47. Sim R, Little JJ (2006) Autonomous vision-based exploration and mapping using hybrid maps and Rao-Blackwellised particle filters.

IEEE Int Conf Intell Robot Syst. https://doi.org/10.1109/IROS.2006.282485
48. Wen S et al (2018) Camera recognition and laser detection based on EKF-SLAM in the autonomous navigation of humanoid robot. J Intell Robot Syst Theory Appl. https://doi.org/10.1007/s10846-017-0712-5
49. Smith R, Self M, Cheeseman P (1988) Estimating uncertain spatial relationships in robotics. Mach Intell Pattern Recognit. https://doi.org/10.1016/B978-0-444-70396-5.50042-X
50. Neira J, Tardós JD (2001) Data association in stochastic mapping using the joint compatibility test. IEEE Trans Robot Autom. https://doi.org/10.1109/70.976019
51. Kato H (2012) Introduction to augmented reality. J Inst Image Inf Telev Eng. https://doi.org/10.3169/itej.66.53
52. Aliyu F, Talib CA (2019) Virtual reality technology. Asia Proc Soc Sci. https://doi.org/10.31580/apss.v4i3.856
53. Milgram P, Kishino F (1994) Taxonomy of mixed reality visual displays. IEICE Trans Inf Syst 77(12):1321–1329
54. Milgram P, Fumio K (2003) A taxonomy of mixed reality visual displays. IEICE Trans Inf Syst 2003
55. Flavián C, Ibáñez-Sánchez S, Orús C (2019) The impact of virtual, augmented and mixed reality technologies on the customer experience. J Bus Res. https://doi.org/10.1016/j.jbusres.2018.10.050
56. Pan Z, Cheok AD, Yang H, Zhu J, Shi J (2006) Virtual reality and mixed reality for virtual learning environments. Comput Graph 30(1):20–28. https://doi.org/10.1016/j.cag.2005.10.004
57. Rokhsaritalemi S, Sadeghi-Niaraki A, Choi SM (2020) A review on mixed reality: current trends, challenges and prospects. Appl Sci (Switzerland). https://doi.org/10.3390/app10020636
58. Mallikarjuna Rao AJ, Sharma M (2017) HoloLens blueprints : experience the virtual and real worlds coming together with HoloLens, 6th ed. Birmingham, England ; Mumbai, India : Packt Publishing. [Online]. Available: https://learning.oreilly.com/library/view/hololens-blueprints/9781787281943/
59. Meulstee JW et al (2019) Toward holographic-guided surgery. Surg Innov. https://doi.org/10.1177/1553350618799552
60. Rubino D, Rubino D (2016) Microsoft HoloLens – Here are the full processor, storage and RAM specs, 2.5.2016. https://docs.microsoft.com/en-us/windows/mixed-reality/hololens-hardware-details. Accessed 04 Apr 2019
61. Vroegop D (2017) Microsoft HoloLens developer's guide. Packt Publishing. [Online]. Available: https://learning.oreilly.com/library/view/microsoft-hololens-developers/9781786460851/
62. Microsoft, MR Basics 100: Getting started with Unity. https://docs.microsoft.com/en-us/windows/mixed-reality/holograms-100
63. Microsoft (2018) HoloLens research mode. https://docs.microsoft.com/en-us/windows/mixed-reality/research-mode
64. Microsoft Hololens Hardware, [Online]. Available: https://docs.microsoft.com/en-us/windows/mixed-reality/hololens-hardware-details
65. Microsoft (2016) Using the HoloLens emulator. https://docs.microsoft.com/en-us/windows/mixed-reality/using-the-hololens-emulator. Accessed 01 Jan 2020
66. Liu Y, Dong H, Zhang L, El Saddik A (2018) Technical evaluation of HoloLens for multimedia: a first look. IEEE Multimed. https://doi.org/10.1109/MMUL.2018.2873473
67. Soares I, Sousa RB, Petry M, Moreira AP (2021) Accuracy and repeatability tests on HoloLens 2 and htc vive. Multimodal Technol Interact. https://doi.org/10.3390/mti5080047
68. Aldebaran Robotics Website (2019) https://www.aldebaranrobotics.com/en/Home/welcome.html?language=en-GB. Accessed 25 Apr 2019
69. Hugel V et al (2009) Mechatronic design of NAO humanoid. pp 769–774. https://doi.org/10.1109/robot.2009.5152516.
70. Aldabarn Robotics (2019) NAO software documentaion
71. Bergmann F (2015) Acoustic communication between two robots based on the NAO robot system, Bachelor Thesis
72. López-Caudana EO, González Gutiérrez CD (2016) Fuzzy PD controller in NAO system's platform. Automation and control trends. InTech. https://doi.org/10.5772/63979
73. Hartley R, Zisserman A, Hartley R, Zisserman A (2011) Camera models. Multiple view geometry in computer vision. Cambridge University Press, Cambridge. https://doi.org/10.1017/cbo9780511811685.010
74. Hartley R, Zisserman A (2004) Multiple view geometry in computer vision. Cambridge University Press, Cambridge. https://doi.org/10.1017/cbo9780511811685
75. Andrew AM (2001) Multiple view geometry in computer vision. Kybernetes. https://doi.org/10.1016/S0143-8166(01)00145-2
76. Zhang H, Zhang C, Yang W, Chen CY (2015) Localization and navigation using QR code for mobile robot in indoor environment. In: 2015 IEEE Int. Conf. Robot. Biomimetics, IEEE-ROBIO 2015, no. March, pp 2501–2506. https://doi.org/10.1109/ROBIO.2015.7419715
77. Girisha H, Dheerendra Kumar A, Singh A, Bharath KP, Deepak (2022) QR code detection. Int J Adv Res Sci Commun Technol. https://doi.org/10.48175/ijarsct-5353
78. Huo L, Zhu J, Singh PK, Pavlovich PA (2021) Research on QR image code recognition system based on artificial intelligence algorithm. J Intell Syst. https://doi.org/10.1515/jisys-2020-0143
79. Alexandrea P (2019) Top 12 Best 3D software for beginners. https://www.3dnatives.com/en/3d-software-beginners100420174/. Accessed 01 Apr 2019