IJRCS

International Journal of Robotics and Control Systems
Vol. 4, No. 4, 2024, pp. 1584-1601
ISSN 2775-2658
http://pubs2.ascee.org/index.php/ijrcs

ASCEE

Accurate Robot Navigation Using Visual Invariant Features and Dynamic Neural Fields

Younès Raoui [a,1,*], Nouzha Elmennaoui [a,2]

[a] Conception & Systems Laboratory, Faculty of Sciences, Mohammed V University in Rabat, Morocco
[1] y.raoui@um5r.ac.ma; [2] n.elmennaoui@um5r.ac.ma
[*] Corresponding Author

ARTICLE INFO

Article History
Received June 15, 2024
Revised August 29, 2024
Accepted September 19, 2024

Keywords
Robot Navigation;
Monocular SLAM;
Visual Feature Points;
Dynamic Neural Fields;
Obstacle Avoidance

ABSTRACT

Robot navigation systems are based on Simultaneous Localization and Mapping (SLAM) and obstacle avoidance. We construct maps for the robot using computer vision methods requiring high repeatability for consistent feature tracking. Also, the obstacle avoidance method needs an efficient tool for fusing data from multiple sensors. This research enhances SLAM accuracy and obstacle avoidance using advanced visual processing and dynamic neural fields (DNF). We propose two key methods: (1) an enhanced multiscale Harris detector using steerable filters for robust feature extraction, achieving around 90% repeatability; and (2) a dynamic neural field algorithm that predicts the optimal heading angle by integrating visual descriptors and LIDAR data. The first method's experimental results show that the new feature detector achieves high accuracy, outperforming existing methods. Its invariance to the orientation of the image makes it insensitive to the rotations of the robot. We applied it to the monocular SLAM and remarked that the positions of the robot were computed precisely. In the second method, the results showed that the dynamic neural fields algorithm ensures efficient obstacle avoidance by fusing the gist of the image and LIDAR data, resulting in more accurate and consistent navigation than laser-only methods. In conclusion, the study presents significant advancements in robot navigation through robust feature detection for SLAM and effective obstacle avoidance using dynamic neural fields. These advancements significantly enhance precision and reliability in robot navigation, paving the way for future innovations in autonomous robotic applications.

1. Introduction

Robot navigation involves two primary approaches: the Centralized Sense-Plan-Act (SPA) architectures and the Distributed Behavior-Based Architectures. In SPA, the robot creates a global representation of the environment, known as a map, using Simultaneous Localization and Mapping (SLAM). In fact, SLAM aims to estimate a mobile robot's trajectory while concurrently constructing a representation of the environment. The robot senses its surroundings to construct an accurate map, plans a set of actions, and then acts based on this plan. Conversely, in the Distributed Behavior-Based Architecture, the robot handles sub-problems separately to create basic behaviors that trigger immediate reactions. This approach produces immediate reactions to stimuli, such as images or LIDAR scans, using dynamic systems like the differential equations of the neural fields developed by Amari [1], predicting the best actions. This paper presents two key innovations: an enhanced mul-

tiscale Harris detector utilizing steerable filters and a dynamic neural field algorithm for efficient obstacle avoidance.

SLAM techniques are essential to compute the robot's poses given the environment represented by a map. The problem we aim to solve is the visual frame matching. The visual features that are not invariant may increase the uncertainty of the computation of the visual landmarks. To solve this, the features should be extracted from the same image but in different fields of view. This will give an invariant descriptor. The second problem we want to tackle is the robot obstacle avoidance. Because of the data the robot acquire from the environment, the actions the robot should take when it finds an obstacle can be errornous. The use of the Dynamic Neural Fields can predict the best actions by taking as input several sources such as images and range data.

Despite advancements in SLAM and obstacle avoidance, challenges remain in achieving high repeatability in feature tracking and robust obstacle detection in dynamic environments. We focus on using visual sensors to build the map. During various SLAM steps, matching between consecutive frames or re-localizing the robot involve comparing perceived landmarks with previous landmarks by matching their local patches. Repeatability is crucial as it ensures consistent feature tracking, enabling reliable tracking of a feature point in multiple frames. Accurate camera pose estimation relies on re-observing the same features insured by robust data association that depends on the repeatability of the feature. These factors highlight the importance of creating an effective detector for feature points and descriptors of the region around them.

Incorporating steerable filters into a Harris multiscale detector provides orientation invariance, enhanced feature detection and computational efficiency. Moreover, Dynamic Neural Fields (DNFs) are particularly useful in scenarios where real-time adaptability, they handle noise and uncertainty in the stimuli signals and can model complex behaviors.

Extraction of visual features: Several visual feature extraction methods are used for visual SLAM applications such as the ORB and BRISC features used in ORBSLAM2 [2]. Their invariance against orientation and scale changes help for computing accurate visual landmarks. DeTone et al. presented SuperPoint, a self-supervised framework for interest point detection and description trained on synthetic data, which shows strong performance in terms of repeatability [3], [5]. Revaud presented R2D2, a feature point detector and descriptor characterized by high repeatability and reliability using a Convolutional Neural Network (CNN) to ensure the robustness of critical points to various transformations [6], [7]. Luo et al. incorporated an attention mechanism to improve the performance of SuperPoint, resulting in ASLFeat [8]. Dusmanu et al. developed an end-to-end trainable CNN for joint detection and description of local features called D2-Net [9]. ContextDesc, conceived by Luo et al., uses a CNN to extract dense feature maps that integrate information from a larger image region surrounding a local feature, helping disambiguate similar local features [10], [13]. Ono et al. developed LF-Net, which predicts the probability of each pixel being a keypoint using a heatmap representation, improving the descriptor's accuracy by applying random homographies to the input image [14].

Visual SLAM: In visual SLAM, the robustness of a system depends on the repeatability of the used feature points. Tareen et al. compared various features such as SIFT, SURF, KAZE, AKAZE, ORB, and BRISK [15]–[17]. Fan et al. developed semantic-aware local features for visual localization by utilizing a CNN architecture for ASLFeat features. Mu et al. presented an algorithm that extracts both line and point features, matching them precisely using the GMS algorithm and RatioTest algorithm [18]. Ruotsalainen et al. improved tracking accuracy in collaborative robotics by using KAZE features for multiscale feature detection with high repeatability [19]. Molina et al. fused data from inertial measurement, optical cameras, and laser range finders to compute a delta-position (DPOS) measurement, improving visual odometry by ensuring invariance to the scale of visual features [20]. Bellavia et al. presented an improved version of HarrisZ called HarrisZ+ that

uses adaptive thresholding to adjust corner detection sensitivity dynamically [7], [21]–[23]. Fontan et al. developed AnyFeature-VSLAM based on ORB-SLAM2, introducing automated parameter tuning to switch between different visual features effortlessly [24], [25].

Navigation with with obstacle avoidance: Reactive navigation based on stimuli also benefits from advanced methods. Orr et al. surveyed deep reinforcement learning (DRL) for multi-robot systems, where more than one robot might be needed to complete a task independently [26]. Advances in attention mechanisms have led to new techniques for detecting obstacles in mobile robots by focusing on relevant features [27], [28]. Schulze discussed a new encoder-based architecture where the encoder improves feature memorability, and a dynamic neural field creates a neural-implicit kinematic model for motion planning [29]. Zhang et al. focused on various neural network models like ZNN, RNN, and GNN for optimal control of redundant manipulators [30]. Li et al. integrated a DRL agent based on a Deep Q network for high-level control commands to avoid dynamic obstacles, employing a predictive controller for reaching target waypoints and rejecting disturbances for Unmanned Surface Vehicles (USVs) [31]–[33]. Chen et al. trained a convolutional neural network to predict actions from a feature map for map-based navigation [34]. Varma et al. described a new path planning approach based on adaptive beetle swarm levy flight optimization (BSLFO) for optimal path calculation, with obstacle avoidance ensured by a deep learning model optimized with a swarm intelligence procedure [35]–[37]. Cimurs developed a goal-oriented navigation algorithm based on Continuous Action Space (CAN) in deep reinforcement learning, using depth information and goal position in polar coordinates [38], [41]. Li et al. extracted the compact obstacle distribution using volumetric representations and employed StereoVoxelNet, a deep neural encoder-decoder to detect occupancy from stereo images [42]. Wenzel et al. used depth maps as inputs to a generative adversarial network for obstacle avoidance [43]. Putra utilized embodied neuromorphic AI by deploying spiking neural networks (SNNs) for obstacle avoidance [44].

In this paper, we address these challenges by presenting a novel feature point detector and a dynamic neural field algorithm for effective obstacle avoidance, contributing to improved navigation in complex environments. The feature detector extracts salient features necessary for matching successive views in the SLAM algorithm. Then the dynamic neural field we used allows to predict the best actions the robot should do in response to stimuli. We make a sensor fusion method to combine stimuli from the camera and the LIDAR.

2. The First Proposed Method: Mapping With Steerable Filters

In the following, we present a method of visual feature extraction that uses steerable filters to compute an invariant feature for robot rotation. To validate the efficiency of the detector and its descriptor, we compute metrics such as repeatability, which gives a good rate of accuracy.

2.1. Observations of the Landmarks

The camera captures the image, and the robot observes the near landmarks on the map eq. (1). By matching, the robot can correct its pose. Then it should project them using the inverse pinhole model to the new keyframe.
Let us have the pose of the landmark y_i in the world frame. The pose of this feature in the camera frame is given with its observation :

$$h_L^R = R^{RW}(y_i^W - r^W) \tag{1}$$

Where R^{RW} is the inverse of the rotation matrix given by transforming the quaternion of the robot state to a rotation matrix, and r^W is the pose of the camera in the world frame.

To get coordinates of the predicted feature, we multiply it by the intrinsic parameters of the right and the left cameras :

For the left camera, the observation is given with eq. (2):

$$h_i = \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} u_{l0} - fkl_u \frac{h^R_{Lx}}{h^R_{Lz}} \\ v_{l0} - fkl_v \frac{h^R_{Ly}}{h^R_{Lz}} \end{bmatrix} \tag{2}$$

Where u_{l0}, v_{l0}, fkl_u and fkl_v are the standard camera calibration parameters. The same equation is for the right camera expect that we use its specific intrinsic parameters.

2.2. The Visual Detector Descriptor

We start by acquiring an image and proceed to filter it using steerable filters across several orientations. From each filtered image, we extract multiscale Harris feature points, which are then clustered. After this initial clustering, we apply a secondary clustering method to further group the feature points, and calculate the mean for each cluster. For each resulting feature point, we create a patch surrounding it. Each patch is then represented in various color spaces. Subsequently, we compute the entropy of the luminance for each patch. Finally, we create a descriptor composed of five floats to encapsulate this information.

2.2.1. The Steerable Filter

The steerable filters are a class of orientation-selective filters used in computer vision whose principle is to steer the filter to the desired angle applied to the image. Mathematically, this operation consists to take a linear combination of a set of basis filters. Given the orientation $\theta \in [0, 2\pi]$ radians. Let's $h(x, y)$ be the steerable filter at the orientation θ applied to the image I(x, y).

$$h(\theta, x, y) = \sum_{i=1}^{N} c_i(\theta) \cdot h_i(x, y) \tag{3}$$

Where $h_i(x, y)$ are the basis filters chosen to be the derivatives of the Gaussians for example. $c_i(\theta)$ are the steering coefficients that depend on the desired orientation θ. N is the number of the steerable filters.

The filter response $F(\theta, x, y)$ at orientation θ can be obtained by convolving the image with the steerable filter:

$$\begin{aligned} F(\theta, x, y) &= I(x, y) * h(\theta, x, y) \\ F(\theta, x, y) &= I(x, y) * \left(\sum_{i=1}^{N} c_i(\theta) \cdot h_i(x, y) \right) \\ F(\theta, x, y) &= \sum_{i=1}^{N} c_i(\theta) \cdot (I(x, y) * h_i(x, y)) \end{aligned} \tag{4}$$

The result is a new image that highlights the features in the original image that aligns with the filters' orientation such as the edge or the texture. Plus, the expression of the coefficients $c_i(\theta)$ can be for example a set of trigonometric functions of the orientation using the scale-dependent derivatives.

2.2.2. Application of the Multiscale Harris Detector With Kmean Clustering

The multiscale Harris detector is an extension of the Harris detector that runs on multiple scales to detect the corners in the image. The mathematical formulation of the multi-scale Harris detector involves computing a multi-scale structure tensor, also known as the second-moment matrix:
Let's $F(x, y, \theta)$ be the input of the corner detector:

$$I_{xx}(\sigma) = \frac{\partial}{\partial x}(G(\sigma) * F) \tag{5}$$

$$I_{yy}(\sigma) = \frac{\partial}{\partial y}(G(\sigma) * F) \tag{6}$$

$$M(\sigma) = \begin{bmatrix} \sum_{u,v} w(u,v;\sigma) I_x^2(\sigma) & \sum_{u,v} w(u,v;\sigma) I_x(\sigma) I_y(\sigma) \\ \sum_{u,v} w(u,v;\sigma) I_x(\sigma) I_y(\sigma) & \sum_{u,v} w(u,v;\sigma) I_y^2(\sigma) \end{bmatrix} \tag{7}$$

The corner points are identified at multiple scales by finding local maxima of R in both scale and space. The multi-scale Harris response function $R(\sigma)$ is calculated for each scale:

$$R(\sigma) = \det(M(\sigma)) - k \cdot (\text{trace}(M(\sigma)))^2 \tag{8}$$

The corner points are identified at multiple scales by finding local maxima of R in both scale and space.

Besides, given that we extract the multiscale Harris feature points from N filtered image, the number of these points become very high. To reduce it, we apply a kmean clustering to create cluster of the most similar points, then we keep the mean of each cluster see Fig. 1.

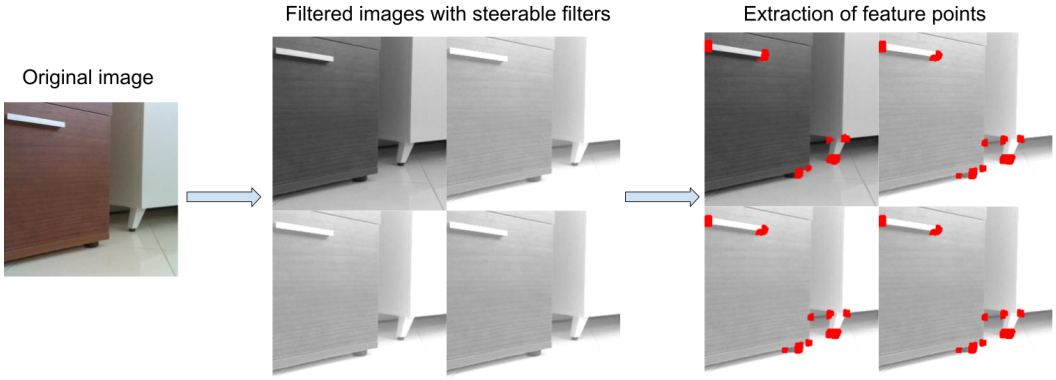


Fig. 1. The feature extraction from each of the image filtered with steerable filters

2.3. The Computation of the Descriptor

We extract a square patch of size 20x20 pixels around each feature point in the image. Then, we create different representations of the patch by quantizing the color information. That's why we generate five different colormap representations of the 20x20 patch. Each colormap corresponds to a different quantization level in the RGB space. The quantization process reduces the number of colors in the patch, which is done by clustering similar colors into discrete bins. Each of the five colormaps quantizes the RGB colors in 16, 32, 64, 128, and 256 channels.

We compute the entropy of each patch by measuring the information content within each patch representation. The entropy measures the randomness or unpredictability of color distributions within the patch.

$$H = -\sum_{i=1}^{n} p_i \log(p_i) \tag{9}$$

Where p_i is the probability of each color level in the quantized colormap. This gives a measure of how diverse the color distribution is within the patch. This yields 5 entropy values shown in Fig. 2.

We formulate a robust descriptor for each feature point based on the computed entropies and their variability.

$$\text{Descriptor} = [H_1, H_2, H_3, H_4, H_5] \tag{10}$$

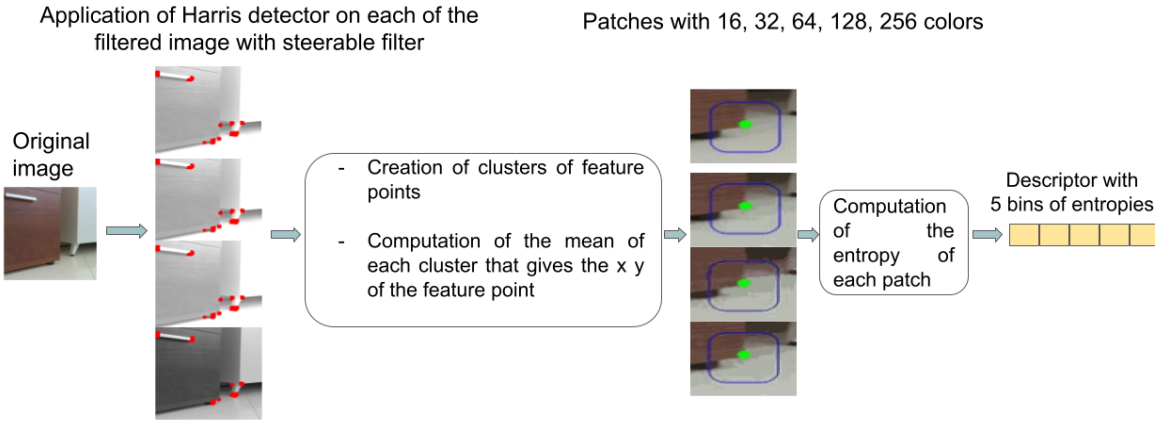


Fig. 2. Schematic representation of the image processing pipeline

2.4. Detector Descriptor Evaluation

To evaluate the multi-scale Harris detector, it's essential to measure its ability to describe interest points across different scales. Several measures can be used:

- Repeatability: This metric assesses how consistently the detector identifies the same points under various transformations, such as rotation and scaling. High repeatability indicates robustness.
- Scale-invariance: The detector should identify features across multiple scales, enabling matching between images with different resolutions.
- Accuracy: This property captures the distinctive characteristics of interest points, reflecting how well the detector differentiates features.

Evaluating these factors determines the detector's effectiveness in applications like Simultaneous Localization and Mapping. In our paper, we propose evaluating our detector using only repeatability due to the importance of viewpoint invariance in robot navigation. It is given with:

$$R = \frac{|F_1 \cap F_2|}{\min(|F_1|, |F_2|)} \tag{11}$$

- $|F_1|$ is the number of features detected in the first image.
- $|F_2|$ is the number of features detected in the second image.
- $|F_1 \cap F_2|$ is the number of features that are consistently detected in both images

Furthermore the recall is given with:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{12}$$

Where TP are the true positives and FN are the false negatives.

2.5. Application to Navigation Methods

Our feature detector can be applied to SLAM methods with traditional navigation algorithms like the A* algorithm or Dynamic Window Approach (DWA) used in ROS (Robot Operating System). By combining its precision with path planning algorithms, we can enhance the robot's ability to navigate in complex environments while avoiding obstacles in real-time [45], [46]. Our feature is more resilient to environment changes. It also improves the SLAM accuracy because the ability to maintain consistent feature tracking across different frames and rotations of the robot reduces drift

and errors in the generated maps, leading to more reliable navigation. Morover, the robustness of the feature detection means that the robot can better interpret complex structures or moving obstacles.

2.6. Monocular SLAM Using our New Detector Descriptor

Our proposed visual feature is used in several steps of the Monocular SLAM. During the initialization, it is used to match features between the first few frames to estimate the initial camera pose. It is also used during the mapping to refine the map because we associate new observations with existing map points. It serves to predict where each feature will appear in the image before searching for it which reduces the computational cost associated with feature matching. We use template matching to match between the descriptor of the projected template patch into the current camera estimate and we scan over the image data until a match is found.

However, the method heavily relies on accurate intrinsic parameters of the cameras. Any calibration errors or variations in camera parameters can significantly impact the accuracy of the pose estimation and feature matching. This sensitivity to calibration can affect the navigation of the robot in dynamic or uncontrolled environments where recalibration might not be feasible.

In the following, we present a short description of the monocular SLAM with inverse depth parametrization.

2.6.1. Inverse Depth Parametrization of the Landmarks

We use the inverse depth parametrization in monocular SLAM to represent accurately the 3D position of the feature. If we use the depth, the measure might be not accurate because when the robot is far from the landmark, the uncertainty increases. For a given feature point observed from a particular viewpoint, the inverse depth parametrization is given with:

$$\mathbf{y} = \begin{bmatrix} x_i \\ y_i \\ z_i \\ \alpha_i \\ \beta_i \\ \rho_i \end{bmatrix} \tag{13}$$

(x_i, y_i and z_i) represents the pose of the landmark where it is first seen with the camera. α and β are the azimut and the elevation that define the bearing of the point from the camera's optical center. ρ is the inverse of the depth.

State Representation: The state of the robot is given with:

$$\mathbf{x} = \begin{bmatrix} \mathbf{p} & \mathbf{q} & y_1 & \cdots & y_n \end{bmatrix} \tag{14}$$

Where p and q are the position and orientation in quaternion of the camera.

2.6.2. Prediction

In the prediction of the Extended Kalman Filter (EKF), The next pose of the robot is predicted by using the motion model of the robot. It gives the current state x_k based on the previous state x_{k-1} and the control of the wheels u_k using a non linear function f.

$$\mathbf{x}_{k|k-1} = f\left(\mathbf{x}_{k-1|k-1}, \mathbf{u}_k\right) \tag{15}$$

The predicted covariance P_k is given with:

$$\mathbf{P}_{k|k-1} = \mathbf{F}_k \mathbf{P}_{k-1|k-1} \mathbf{F}_k^\top + \mathbf{Q}_k \tag{16}$$

Here, F_k is the Jacobian of the motion model with respect to the state, and Q_k is the process noise covariance.

2.6.3. Update

In the update of the EKF with the inverse depth parametrization, the measurement model converts the inverse depth parametrization into a pixel measurement. The state estimation of the measure is given with:

$$z_{ki} = h(x_{k|k-1}, y_i) + v_k \tag{17}$$

z_k is the state of the observation. $x_{k|k-1}$ is the probabilistic transition model. y_i is the inverse depth state of the -th feature. v_k is the model gaussian noise.

To get z_k, let's project the 3D feature y_i with the camera that is in the pose (p, q), where p is the position and q is the orientation. First we transform the feature point from the inverse depth coordinate to the camera-centered coordinate:

$$\mathbf{X}_c = \frac{1}{\rho_i} \begin{bmatrix} \cos(\beta_i)\sin(\alpha_i) \\ \sin(\beta) \\ \cos(\beta_i)\cos(\alpha_i) \end{bmatrix} \tag{18}$$

Plus, we rotate and translate the point to the world frame using the camera pose:

$$\mathbf{X}_w = \mathbf{R}(\mathbf{q})\mathbf{X}_c + \mathbf{p} \tag{19}$$

Where R(q) is the rotation matrix corresponding to the quaternion.

Then we project the world frame coordinates onto the image plane using the camera's intrinsic parameters (focal lengths f_x, f_y and optical center c_x, c_y).

$$\mathbf{z}_k = \begin{bmatrix} f_x \frac{X_w}{Z_w} + c_x \\ f_y \frac{Y_w}{Z_w} + c_y \end{bmatrix} \tag{20}$$

Where X_w, Y_w, Z_w are the components of $\mathbf{X}_w$.

$$\mathbf{H}_y = \frac{\partial \mathbf{z}_k}{\partial \mathbf{y}} = \frac{\partial \mathbf{z}_k}{\partial \begin{bmatrix} \alpha \\ \beta \\ \rho \end{bmatrix}} \tag{21}$$

The update of the robot position and the landmarks is given with:

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^\top (\mathbf{H}_k \mathbf{P}_{k|k-1}\mathbf{H}_k^\top + \mathbf{R}_k)^{-1} \tag{22}$$

$$\mathbf{x}_{k|k} = \mathbf{x}_{k|k-1} + \mathbf{K}_k(\mathbf{z}_k - h(\mathbf{x}_{k|k-1})) \tag{23}$$

K_k is the Kalman gain, H_k is the jacobian of the measurement. R_k is the measurement noise covariance. The updated covariance is given with:

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_{k|k-1} \tag{24}$$

2.6.4. Data Association

It determines the correspondance between the observed feature z_i and the map landmarks x_{m_j} with the Mahalanobis distance that measures the similarity between the observed features and the predicted measurements of landmarks. The innovation r_i^j is given with:

$$\mathbf{r}_i^j = \mathbf{z}_i - \mathbf{h}(p, y_j) \tag{25}$$

and the mahalanobis distance is given with:

$$d_i^j = \sqrt{(\mathbf{r}_i^j)^{\mathrm{T}} \mathbf{S}_i^{-1} \mathbf{r}_i^j} \tag{26}$$

and:

$$\mathbf{S}_i = \mathbf{H}_i \mathbf{P} \mathbf{H}_i^{\mathrm{T}} + \mathbf{R}_i \tag{27}$$

R_i is the measurement noise covariance matrix. An observed feature z_i is associated with the landmark y_j is the Mahalanobis distance d_i^j is lower than a threshold τ.

When the feature is repeatable, it means that the same physical feature is detected as z_i in different frames. It is crucial when calculating the innovation as it relies to the observed and predicted features. Also when the feature is repeatable, the Mahalanobis distance will be lower. Plus, it insures that the covariance matrix S_i which involves the measurement noise covariance R_i accurately reflects the uncertainty in the measurements.

Besides the inverse depth parametrization of the landmarks may lead to potential errors in depth estimation when converting between different coordinate systems and managing the associated uncertainties. Particularly when in environments with rapidly changing scenes or fast-moving objects.

3. The Second Proposed Method: Navigation with the Dynamic Neural Fields

We present the second method, which consists of developing a technique for robot navigation that learns using a dynamic neural field. It receives visual stimuli and the distance and orientation from the obstacle as inputs. Then, using a maximization algorithm based on the activations of the neural field, we learn the best action to perform. We use the Euclidean distance between the desired and the computed pose obtained by taking the predicted action to evaluate the technique see in Fig. 3.

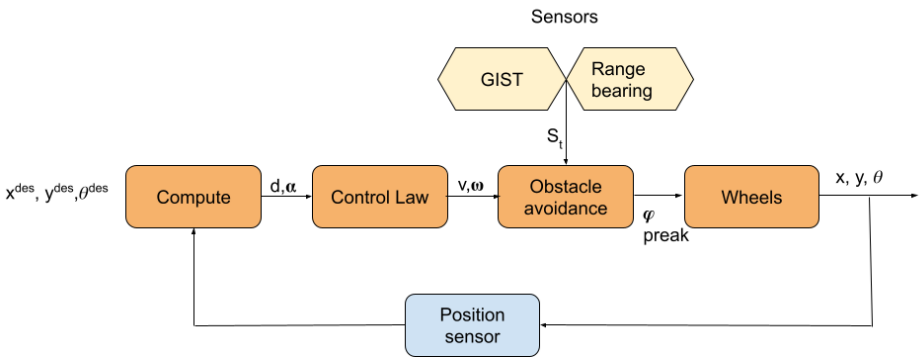


Fig. 3. Block diagram of the reactive navigation system incorporating the gist and range-bearing sensors. The system computes the desired position and orientation, applies control laws, avoids obstacles, and controls the wheels based on sensor inputs and the current position

3.1. Overview of the Dynamic Neural Fields

The Dynamic Neural Fields (DNF) is a neural network architecture used to make decisions of the state of a non linear dynamic system. If we have N states of a system, we consider N populations of neurons with activities $A_k = g(u_k)$ where g is a sigmoid for example. DNF are used in many robotic applications such as graspoing and behavior generation. Also they are used to implement the visual attention and to model time series [47]–[50].

The DNF are modeled with the Amari equation that predicts the best policy to take after receiving noisy stimuli that could be for example a bell curve. It models the activations of each neuron with the others by computing the inhibitory connexions. Then we find the neuron with the maximum activation that gives us the best heading direction.

$$\tau \dot{u}(\varphi, t) = -u(\varphi, t) + S(\varphi, t) + h + \int_{-\infty}^{+\infty} w(\varphi, \dot{\varphi}) f(u(\dot{\varphi}, t)) d\dot{\varphi} \tag{28}$$

$u(\varphi, t)$ is the firing potential, h is the resting activity level of the neuron. $w(\varphi, \dot{\varphi})$ is a bell shaped kernel that represents local excitatory and global inhibitory inflow from adjacent connections in the field.

- If $u(\varphi) \leq 0, f(u(\dot{\varphi}, t)) = 0$ then $\tau \dot{u}(\varphi, t) = -u(\varphi, t) + S(\varphi, t) + h$
- If the stimulus isn't sufficient, then the system remains inactive or below the threshold.
- If $u(\varphi) > 0$, the system doesn't have a specific pic, $f(u(\dot{\varphi}, t)) \neq 0$ for all φ.
- If $\exists \varphi, a_1 \leq \varphi \leq a_2, u(\varphi) > 0$, the excitation u is localized and moves in a stationary trajectory in the system (28). The solution is characterized by a single peak that indicates a focused area of the activity.

The solution $u(\varphi, t)$ shows a competition between N populations A_i in the response of the external inputs that creates a dynamic interaction between the exitatory populations through the common pool of inhibitory neurons mediated with the weights W_{EE}, W_{EI} and W_{IE}. The DNF shows a competition between exitatory and inhibitory populations through a shared inhibitory pool.

The neural architecture that implements the DNF is named winner-take-all network that makes a decision on the winner between N populations.

$$\varphi_{\text{peak}} = \arg_{\max}\{u(\varphi) \mid \varphi \in [1, N]\} \tag{29}$$

It is worth mentioning that The differential equation of the DNF and the interaction weights require careful tuning, which can be challenging and time-consuming. The choice of the parameters of the DNF model can significantly impact the system's performance which requires fine-tuning. Morover, the accuracy of integrating the range and bearing stimulus into the DNF depends on the precision of the Lidar or laser range finder sensors. This accuracy can adversely affect the navigation performance in cluttered or dynamic obstacle environments.

3.2. Application to the Robot Navigation

The robot navigation is made with two behaviors : target acquisition and obstacle avoidance. We give to the robot the desired trajectory that is transformed with the control law to linear and angular velocities. However we should update those velocities to avoid obstacles and reach the target. We use the DNF in order to decode the correct heading direction given the stimulus S_t.

$$\tau \dot{u}(\varphi, t) = -u(\varphi, t) + S(\varphi, t) + h + \sum_{\dot{\varphi}=1}^{N} w(\varphi, \dot{\varphi}) f(u(\dot{\varphi}, t)) \tag{30}$$

We use the following interaction weights :

$$w(\varphi,\dot{\varphi}) = \begin{cases} 1 & \text{if } \|\varphi,\dot{\varphi}\| \leq 3 \\ -2.2 & \text{otherwise} \end{cases} \tag{31}$$

3.2.1. Computation of the Stimulus

Each population of the neurons of the DNF receives the inputs. As we have N head directions, we consider that each population encode one head direction. The obstacles stimulus is given with:

$$\begin{gathered} S_{\text{obstacle}}(\varphi,t) = \sum_{l=1}^{N_{\text{obst}}} g(d_{ol})\, S_{ol}(\varphi,t) \\ S_{ol}(\varphi,t) = c_o e^{-\sigma(\varphi-\varphi_l)^2} \end{gathered} \tag{32}$$

Where φ is the current head direction and φ_l is the direction to the obstacle l. σ is the rang of the inhibition of the obstacle, c_0 is the amplitude of the obstacle stimulus. Furthermore,

$$S_{\text{target}}(\varphi,t) = m_1 - m_2 \cdot \left\|\varphi_{\text{target}},\varphi\right\| \tag{33}$$

φ_{target} is the head direction of the target. The resultant stimulus is given with:

$$S(\varphi,t) = S_{\text{target}}(\varphi,t) - S_{\text{obstacle}}(\varphi,t) \tag{34}$$

Let φ_F be the head direction solution of the DNF differential equation that is the most stable fixed point of it. The target velocity is given with:

$$V_T(t) = V_{\max}\left(1 - e^{-\sigma_v d_T}\right) \tag{35}$$

d_T is the distance between the robot and the target. $\sigma_v > 0$. The resultant obstacle velocity is given with:

$$V_O(t) = C_O\left(1 - g(d_{no})\, e^{-\sigma_{Ov}(\varphi_F - i_{no})^2}\right) \tag{36}$$

C_0, σ_{Ov}¿ 0. d_{no} is the distance of the obstacle relative to the robot. i_{no} is the nearest obstacle direction to the robot movement direction.
The resultant velocity is given with:

$$V(t) = \begin{cases} V_T(t)V_O(t), & \varphi_F > 0 \\ 0, & \varphi_F \leq 0 \end{cases} \tag{37}$$

3.2.2. The Visual GIST Descriptor Stimulus

We describe each scene that the robot perceives by using the gist descriptor developed in [51]. The GIST attempts to describe the spatial structure of the image by looking for the frequencies in the sub-images. Let I(x, y) be the current image.
Processing: We resize the image to have M rows and N columns: $I_{\text{resized}} = \text{resize}(I, M, N)$
Filtering: We filter it with a Gabor filter: λ is the wavelength of the sinusoidal factor, θ is the orientation, ϕ is the phase offset, γ is the spatial aspect ratio, and σ is the standard deviation of the Gaussian envelope:

$$I_{\text{filtered}}^{(\lambda,\theta)} = I_{\text{resized}} * G_{\lambda,\theta,\phi,\gamma}(\sigma) \tag{38}$$

Where * denotes the convolution operation
Windowing: We then create a grid of K*K windows W_k

$$W_k = I_{\text{filtered}}^{(\lambda,\theta)}[x_k : x_{k+1}, y_k : y_{k+1}] \tag{39}$$

Where x_k, x_{k+1}, y_k, y_{k+1} define the coordinates of the k-th windows.

$$F_k^{(\lambda,\theta)} = \frac{1}{|W_k|} \sum_{(x,y)\in W_k} I_{\text{filtered}}^{(\lambda,\theta)}(x,y) \tag{40}$$

Descriptor Formation: We concatenate the average filter responses from all windows and all filters to form the GIST descriptor:

$$\text{GIST} = \left[F_1^{(\lambda_1,\theta_1)}, F_2^{(\lambda_1,\theta_1)}, \ldots, F_{K^2}^{(\lambda_1,\theta_1)}, \ldots, F_1^{(\lambda_L,\theta_M)}, \ldots, F_{K^2}^{(\lambda_L,\theta_M)}\right] \tag{41}$$

Where L is the number of scales and M is the number of orientations used for the Gabor filters. The actual GIST descriptor is a vector that concatenates all these average responses, capturing the dominant spatial structure of the scene. The specific parameters used in the Gabor filter, the number of scales, orientations, and the size of the windows are set in the program.

3.3. The Range and Bearing Stimulus

The range and bearing stimulus is computed from the Lidar or the laser range finder sensors. Let's have the distance d_t and the direction ϕ_t from the robot to an obstacle in the map. The position of the robot is (x_r, y_r) and the position of the obstacle is (x_o, y_o).
The relative position of the obstacle with respect to the robot is:

$$\Delta x = x_0 - x_r \tag{42}$$
$$\Delta y = y_0 - y_r \tag{43}$$

The range is computed as the euclidean distance between the robot and the obstacle:

$$d_{ol} = \sqrt{\Delta x^2 + \Delta y^2} \tag{44}$$

The bearing is the angle between the robot's heading ϕ_r and the line connecting the robot and the obstacle. It is normalized to be between $[-\pi, \pi]$ or $[0, 2\pi]$ radians.

$$\varphi = arctan2(\Delta y, \Delta x) - \phi_r \tag{45}$$

The stimulus is computed with the equation (32).

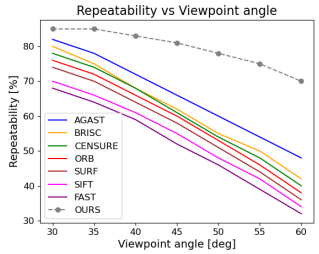
4. Results and Discussion

We did the experiments using Matlab from Mathwork software with the image processing toolbox. The colored images are extracted from indoor environment see Fig. 1 using an AOK-D camera. We present here the results of our two contributions.

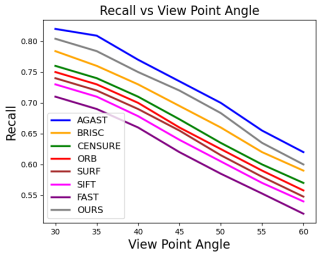
4.1. Results of the Feature Points Detection

In the Fig. 4a, our feature points maintain high repeatability (around 90%) indicating strong performance in identifying the same features despite changes in orientation. Also in the Fig. 4b, the recall of our detector is high comparing to the other features. This behavior illustrates the effectiveness of using steerable filters for orientation invariance. We compare to various feature descriptors (AGAST, BRISC, CENSURE, ORB, SURF, SIFT, FAST). Moreover, the Fig. 4c shows the predicted positions of a robot along with the associated uncertainties (covariances), highlighting areas of higher or lower confidence in the robot's pose estimation within a defined operational space. We deduce that the

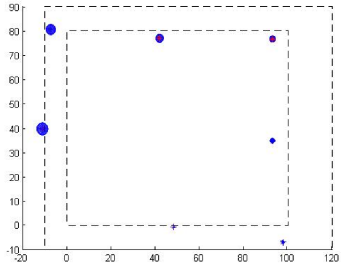
confidence increases when the robot is far from its initial pose. Also, the Fig. 4d shows the estimated positions of the robot. The Fig. 4e and Fig. 4f represent two scenes from a robot SLAM (Simultaneous Localization and Mapping) operation. The landmarks identified by the robot are shown as red dots, while the blue ellipses around them represent the covariance of the landmark poses, indicating the uncertainty in their estimated positions.



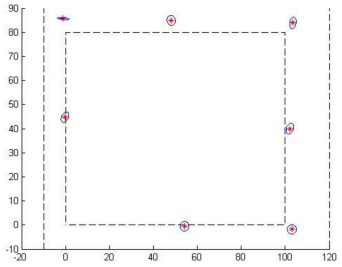
(a) The repeatability score



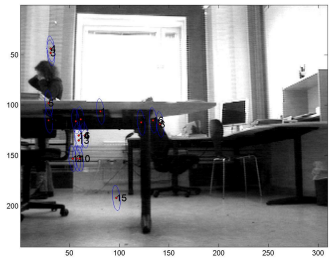
(b) The recall of the detector



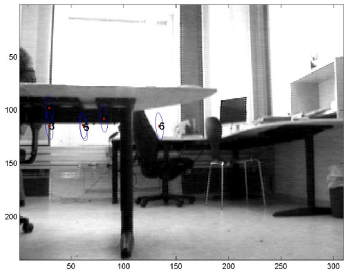
(c) The predicted poses



(d) The estimated poses of the robot after applying the monocular SLAM



(e) The scene 1 with the poses of the landmarks in the image with their covariances



(f) The scene 2 with the poses of the landmarks in the image with their covariances

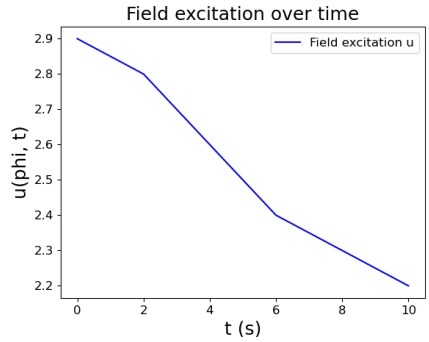
Fig. 4. The result of the new feature points

We conclude that high repeatability is essential for accurate visual SLAM techniques. The fact that our method outperforms other well-known features suggests that changing the orientation of the acquired images with steerable filters is an efficient technique for perception in robot navigation. Besides, Reducing the uncertainty of the robot localization has real-life applications in making decisions because as the robot moves, the uncertainty grows, and it needs to make real-time decisions to mitigate this. It might slow down to reduce the risk of collisions, increase the frequency of sensor data processing, or return to a previously known position to recalibrate its pose.

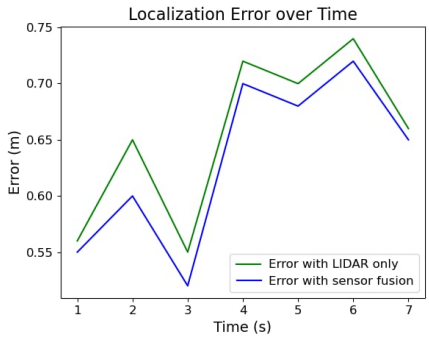
4.2. Results of the Dynamic Neural Fields

The Fig. 5a shows the neural field when it responds to a significant visual-laser stimulus. It rapidly adjusts, and then stabilizes its activity, maintaining a single-peak configuration as described by the neural field theory. The Figs. 5b, 5c, 5d show the standard deviation of the error between the pose of the robot using the neural field and the ground truth with and without the sensor fusion. The fusion method generally has a lower standard deviation, indicating more accurate and consistent performance compared to the laser-only method.

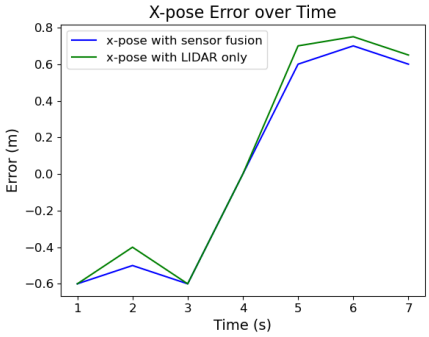
The Fig. 6 shows the robot's motion path while avoiding obstacles (blue circles) using neural fields stimulated by laser range data. The robot's trajectory (marked by triangles) adjusts in real-time to navigate around obstacles, ensuring smooth and efficient movement within the defined area. From the second technique, we conclude that the more the DNF receives several types of stimuli, the more the localization is accurate. We also deduce that the DNF's activations depend on the type of stimuli. This can be justified by the fact that the DNF uses Hebbian unsupervised learning, which is more



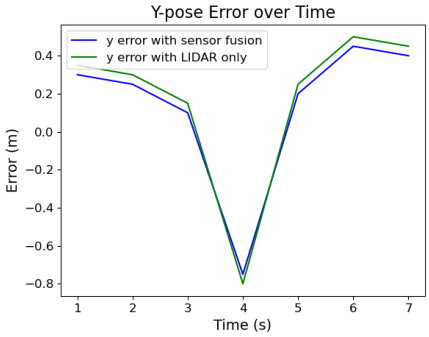
(a) Time evolution of the field excitation u as a function of time t.



(b) Comparison of localization error in the x and y coordinates over time



(c) Comparison of localization error in the x coordinates over time



(d) Comparison of localization error in the y coordinates over time

Fig. 5. The results of the navigation with DNF

reliable if the sources are of different types; in our case, the gist, the range, and the bearing.

5. Conclusion

This paper presents two new techniques that ameliorate robot navigation. In the first technique, we propose a novel feature point detector and visual descriptor applied to monocular SLAM. In the second technique, we propose an algorithm that controls the robot's motion with dynamic neural fields that receive the GIST visual descriptor and the LIDAR scans.

We proved that our first method provides repeatable visual feature points that ameliorate the monocular SLAM's precision through steerable filters that ensure the invariance to the changes in the orientations. Moreover, using several color spaces in the computation of the descriptor results in a good matching of visual points, which is proven by the decrease of the uncertainty in calculating the poses. Furthermore, in the second contribution, we showed that the fusion of both the GIST visual descriptor and the LIDAR scans reduces the positioning error caused by detecting an obstacle.

These techniques represent a meaningful advancement in the field of robot navigation, offering robust and efficient solutions that are applicable to a wide range of real-time scenarios such as autonomous vehicles, drones, and robotic assistants for high speed navigation and efficient processing.

Furthermore, the research results and application prospects can be ameliorated with a pre-trained sequential model instead of the dynamic neural fields to generate the robot trajectory [52]–[54] and use of the CNN for developing real 3D invariant visual features.

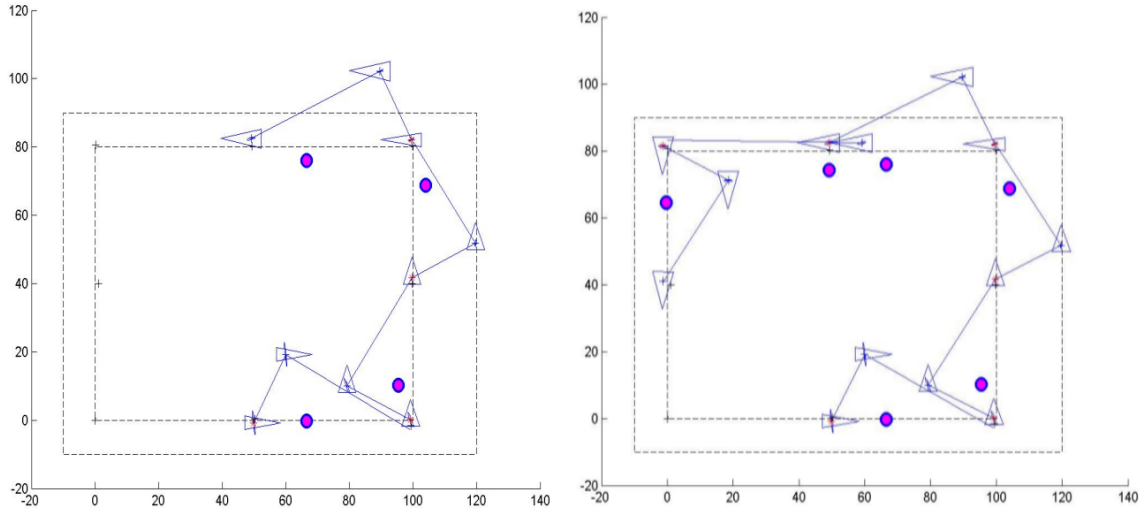


Fig. 6. The navigation with obstacle avoidance with the DNF in two times, the robot is represented with a triangle and the obstacles are represented with circles

Author Contribution: All authors contributed equally to the main contributor to this paper. All authors read and approved the final paper.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

[1] S.-i. Amari, "Dynamics of pattern formation in lateral-inhibition type neural fields," *Biological Cybernetics*, vol. 27, pp. 77–87, 1977, https://doi.org/10.1007/BF00337259.

[2] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras," in *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255-1262, 2017, https://doi.org/10.1109/TRO.2017.2705103.

[3] D. DeTone, T. Malisiewicz and A. Rabinovich, "SuperPoint: Self-Supervised Interest Point Detection and Description," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 337-33712, 2018, https://doi.org/10.1109/CVPRW.2018.00060.

[4] M. Dusmanu *et al.*, "D2-Net: A Trainable CNN for Joint Description and Detection of Local Features," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8084-8093, 2019, https://doi.org/10.1109/CVPR.2019.00828.

[5] A. F. de Araújo, "Deep image features for instance-level recognition and matching," *Proceedings of the 2nd Workshop on Structuring and Understanding of Multimedia heritAge Contents*, 2020, https://doi.org/10.1145/3423323.3423414.

[6] J. Revaud, P. Weinzaepfel, C. R. De Souza, N. Pion, G. Csurka, Y. Cabon, and M. Humenberger, "R2D2: Repeatable and reliable detector and descriptor," in *Advances in Neural Information Processing Systems*, pp. 12405–12415, 2019, https://arxiv.org/pdf/1906.06195.

[7] D. Mishkin, F. Radenovic, and J. Matas, "Repeatability is not enough: Learning affine regions via discriminability," in *European Conference on Computer Vision*, pp. 287–304, 2018, https://doi.org/10.1007/978-3-030-01240-3_18.

[8] Z. Luo, L. Zhou, X. Bai, Y. Yao, J. Li, Z. Hu, M. Chai, and L. Quan, "Aslfeat: Learning local features of accurate keypoint detection and matching," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6589–6598, 2020,

https://openaccess.thecvf.com/content_CVPR_2020/papers/Luo_ASLFeat_Learning_Local_Features_of_Accurate_Shape_and_Localization_CVPR_2020_paper.pdf.

[9] M. Dusmanu *et al.*, "D2-Net: A Trainable CNN for Joint Description and Detection of Local Features," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8084-8093, 2019, https://doi.org/10.1109/CVPR.2019.00828.

[10] Z. Luo *et al.,* "ContextDesc: Local Descriptor Augmentation With Cross-Modality Context," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2522-2531, 2019, https://doi.org/10.1109/CVPR.2019.00263.

[11] F. Darmon, M. Aubry, and P. Monasse, "Learning to guide local feature matches," *2020 International Conference on 3D Vision (3DV)*, pp. 1127–1136, 2020, https://doi.org/10.1109/3DV50981.2020.00123.

[12] J. Corsetti, D. Boscaini, and F. Poiesi, "Revisiting fully convolutional geometric features for object 6d pose estimation," *2023 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pp. 2095–2104, 2023, https://doi.org/10.1109/ICCVW60793.2023.00224.

[13] Q. Chang, Q. Liu, X. Yang, Y. Huang, F. Ren, and Y. Cui, "The relocalization of slam tracking based on spherical cameras," *IEEE Access*, vol. 9, pp. 159764–159783, 2021, https://doi.org/10.1109/ACCESS.2021.3130928.

[14] Y. Ono, E. Trulls, P. Fua, and K. M. Yi, "Lf-net: Learning local features from images," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 6234–6244, 2018, https://arxiv.org/pdf/1805.09662.

[15] S. A. K. Tareen and Z. Saleem, "A comparative analysis of sift, surf, kaze, akaze, orb, and brisk," *2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, pp. 1–10, 2018, https://doi.org/10.1109/ICOMET.2018.8346440.

[16] D. A. Jatmiko and S. U. Prini, "Study and performance evaluation binary robust invariant scalable keypoints (brisk) for underwater image stitching," *IOP Conference Series: Materials Science and Engineering*, vol. 879, 2020, https://doi.org/10.1088/1757-899X/879/1/012111.

[17] R. Rahmania, M. S. Anggreainy, I. H. Kartowisastro, and W. Budiharto, "Identification of clustering brisk keypoint feature in grocery product using the elbow method," *2024 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT)*, pp. 106–111, 2024, https://doi.org/10.1109/IAICT62357.2024.10617640.

[18] Q. Mu, Y. Wang, S. Guo, and Z. Li, "Indoor visual odometry algorithm based on adaptive feature fusion," *2022 2nd International Conference on Algorithms, High Performance Computing and Artificial Intelligence (AHPCAI)*, pp. 121–127, 2022, https://doi.org/10.1109/AHPCAI57455.2022.10087841.

[19] L. Ruotsalainen, A. Morrison, M. Mäkelä, J. Rantanen, and N. Sokolova, "Improving computer vision-based perception for collaborative indoor navigation," *IEEE Sensors Journal*, vol. 22, no. 6, pp. 4816–4826, 2022, https://doi.org/10.1109/JSEN.2021.3106257.

[20] G. D. Molina, M. Hansen, J. Getchius, R. S. Christensen, J. A. Christian, S. M. Stewart, and T. Crain, "Aas 22-113 visual odometry for precision lunar landing," 2022, https://www.semanticscholar.org/paper/AAS-22-113-VISUAL-ODOMETRY-FOR-PRECISION-LUNAR-Molina-Hansen/fdb89e10da6e2587f5701e4d693958cc245f0353.

[21] F. Bellavia and D. Mishkin, "Harrisz$^+$: Harris corner selection for next-gen image matching pipelines," *Pattern Recognit. Lett.*, vol. 158, pp. 141–147, 2022, https://doi.org/10.1016/j.patrec.2022.04.022.

[22] E. Riba, D. Mishkin, D. Ponsa, E. Rublee, and G. R. Bradski, "Kornia: an open source differentiable computer vision library for pytorch," *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 3663–3672, 2020, https://doi.org/10.1109/WACV45572.2020.9093363.

[23] Y. Jin, D. Mishkin, A. Mishchuk, J. Matas, P. Fua, K. M. Yi, and E. Trulls, "Image matching across wide baselines: From paper to practice," *International Journal of Computer Vision*, vol. 129, pp. 517–547, 2021, https://doi.org/10.1007/s11263-020-01385-0.

[24] A. Fontan, J. Civera, and M. Milford, "Anyfeature-vslam: Automating the usage of any feature into visual slam," in *Robotics: Science and Systems (RSS)*, 2024, https://doi.org/10.15607/rss.2024.xx.084.

[25] R. Elvira, J. D. Tardós, and J. M. M. Montiel, "Orbslam-atlas: a robust and accurate multi-map system," *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6253–6259, 2019, https://doi.org/10.1109/IROS40897.2019.8967572.

[26] J. Orr and A. Dutta, "Multi-agent deep reinforcement learning for multi-robot applications: A survey," *Sensors*, vol. 23, no. 7, p. 3625, 2023, https://doi.org/10.3390/s23073625.

[27] K. Sviatov, A. Miheev, S. Sukhov, Y. Lapshov, and S. Rapp, "Detection of obstacle features using neural networks with attention in the task of autonomous navigation of mobile robots," in *Computational Science and Its Applications – ICCSA 2020*, pp. 1013–1026, 2020, https://doi.org/10.1007/978-3-030-58817-5_72.

[28] S. S. Gu, E. Holly, T. P. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates," *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3389–3396, 2017, https://doi.org/10.1109/ICRA.2017.7989385.

[29] L. Schulze and H. Lipson, "High-degrees-of-freedom dynamic neural fields for robot self-modeling and motion planning," pp. 3064–3070, 2024, https://doi.org/10.1109/ICRA57147.2024.10611047.

[30] J. Zhang, L. Jin, T. Deng, and M. Tahir, "Editorial: Learning and control in robotic systems aided with neural dynamics," *Frontiers in Neurorobotics*, vol. 17, 2023, https://doi.org/10.3389/fnbot.2023.1193119.

[31] J. Li, J. Chavez-Galaviz, K. Azizzadenesheli, and N. Mahmoudian, "Dynamic obstacle avoidance for usvs using cross-domain deep reinforcement learning and neural network model predictive controller," *Sensors*, vol. 23, no. 7, p. 3572, 2023, https://doi.org/10.3390/s23073572.

[32] O. Bastani and S. Li, "Safe reinforcement learning via statistical model predictive shielding," *Robotics: Science and Systems XVII*, 2021, https://doi.org/10.15607/RSS.2021.XVII.026.

[33] H. Xing and L. Yu, "Target-driven multi-input mapless robot navigation with deep reinforcement learning," *Journal of Physics: Conference Series*, vol. 2513, 2023, https://doi.org/10.1088/1742-6596/2513/1/012004.

[34] G. Chen, L. Pan, Y. Chen, P. Xu, Z. Wang, P. Wu, J. Ji, and X. Chen, "Deep reinforcement learning of map-based obstacle avoidance for mobile robot navigation," *SN Computer Science*, vol. 2, no. 417, 2021, https://doi.org/10.1007/s42979-021-00817-z.

[35] K. N. V. S. Varma and S. L. Kumari, "Robotic vision based obstacle avoidance for navigation of unmanned aerial vehicle using fuzzy rule based optimal deep learning model," *Evolutionary Intelligence*, vol. 17, pp. 2193–2212, 2024, https://doi.org/10.1007/s12065-023-00881-9.

[36] B. Wang, Y. Sun, N. Zhao and G. Gui, "Learn to Coloring: Fast Response to Perturbation in UAV-Assisted Disaster Relief Networks," in *IEEE Transactions on Vehicular Technology*, vol. 69, no. 3, pp. 3505-3509, 2020, https://doi.org/10.1109/TVT.2020.2967124.

[37] K.-T. Wei and B. Ren, "A method on dynamic path planning for robotic manipulator autonomous obstacle avoidance based on an improved rrt algorithm," *Sensors*, vol. 18, no. 2,p. 571, 2018, https://doi.org/10.3390/s18020571.

[38] R. Cimurs, J. H. Lee, and I. H. Suh, "Goal-oriented obstacle avoidance with deep reinforcement learning in continuous action space," *Electronics*, vol. 9, no. 3, p. 411, 2020, https://doi.org/10.3390/electronics9030411.

[39] Z. Chu, F. Wang, T. Lei and C. Luo, "Path Planning Based on Deep Reinforcement Learning for Autonomous Underwater Vehicles Under Ocean Current Disturbance," in *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 1, pp. 108-120, 2023, https://doi.org/10.1109/TIV.2022.3153352.

[40] J. Cai, A. Du, X. Liang, and S. Li, "Prediction-based path planning for safe and efficient human-robot collaboration in construction via deep reinforcement learning," *Journal of Computing in Civil Engineering*, vol. 37, no.1, 2023, https://doi.org/10.1061/(ASCE)CP.1943-5487.0001056.

[41] C. Yan, G. Chen, Y. Li, F. Sun, and Y. Wu, "Immune deep reinforcement learning-based path planning for mobile robot in unknown environment," *Applied Soft Computing*, vol. 145, p. 110601, 2023, https://doi.org/10.1016/j.asoc.2023.110601.

[42] H. Li, Z. Li, N. Ü. Akmandor, H. Jiang, Y. Wang and T. Padır, "StereoVoxelNet: Real-Time Obstacle Detection Based on Occupancy Voxels from a Stereo Camera Using Deep Neural Networks," *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4826-4833, 2023, https://doi.org/10.1109/ICRA48891.2023.10160924.

[43] P. Wenzel, T. Schön, L. Leal-Taixé, and D. Cremers, "Vision-based mobile robotics obstacle avoidance with deep reinforcement learning," *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 14360–14366, 2021, https://doi.org/10.1109/ICRA48506.2021.9560787.

[44] R. V. W. Putra, A. Marchisio, F. Zayer, J. Dias, and M. Shafique, "Embodied neuromorphic artificial intelligence for robotics: Perspectives, challenges, and research development stack," *ArXiv*, 2024, https://doi.org/10.48550/arXiv.2404.03325.

[45] N. Adiuku, N. P. Avdelidis, G. Tang, and A. Plastropoulos, "Improved hybrid model for obstacle detection and avoidance in robot operating system framework (rapidly exploring random tree and dynamic windows approach)," *Sensors*, vol. 24, no. 7, p. 2262, 2024, https://doi.org/10.3390/s24072262.

[46] J. Zhao, S. Liu, and J. Li, "Research and implementation of autonomous navigation for mobile robots based on slam algorithm under ros," *Sensors*, vol. 22, no. 11, p. 4172, 2022, https://doi.org/10.3390/s22114172.

[47] R. Grieben, J. P. Spencer, and G. Schöner, "Visual selective attention: Priority is all you need," in *Proceedings of the Annual Meeting of the Cognitive Science Society*, vol. 46, 2024, https://doi.org/10.1037/rev0000245.

[48] S. Sehring, R. Koebe, S. Aerdker, and G. Schöner, "A neural dynamic model autonomously drives a robot to perform structured sequences of action intentions," in *Proceedings of the Annual Meeting of the Cognitive Science Society*, vol. 46, 2024, https://escholarship.org/uc/item/3gh831nw.

[49] D. Sabinasz, M. Richter, and G. Schöner, "Neural dynamic foundations of a theory of higher cognition: the case of grounding nested phrases," *Cognitive Neurodynamics*, vol. 18, pp. 557–579, 2024, https://doi.org/10.1007/s11571-023-10007-7.

[50] S. Kamkar, H. A. Moghaddam, R. Lashgari, and W. Erlhagen, "Brain-inspired multiple-target tracking using dynamic neural fields," *Neural Networks*, vol. 151, pp. 121–131, 2022, https://doi.org/10.1016/j.neunet.2022.03.026.

[51] A. Oliva and A. Torralba, "Modeling the shape of the scene: a holistic representation of the spatial envelope," *International Journal of Computer Vision*, vol. 42, no. 3, pp. 145–175, 2001, https://doi.org/10.1023/A:1011139631724.

[52] P. Kao, M. Chahine, A. Ray, M. Lechner, A. Amini, and D. Rus, *Robust flight navigation with liquid neural networks*, Doctoral dissertation, Massachusetts Institute of Technology, 2022.

[53] M. Chahine, R. Hasani, P. Kao, A. Ray, R. Shubert, M. Lechner, A. Amini, and D. Rus, "Robust flight navigation out of distribution with liquid neural networks," *Science Robotics*, vol. 8, no. 77, 2023, https://doi.org/10.1126/scirobotics.adc8892.

[54] C. Vorbach, R. Hasani, A. Amini, M. Lechner, and D. Rus, "Causal navigation by continuous-time neural networks," *arXiv*, 2021, https://arxiv.org/pdf/2106.08314.