MIXO: Mixture of experts-based visual odometry for multicamera autonomous systems

MIXO: Mixture of Experts-based Visual Odometry for Multicamera Autonomous Systems

Lia Morra, *Senior Member, IEEE,* Andrea Biondo, Nicola Poerio, Fabrizio Lamberti, *Senior Member, IEEE*

***Abstract*—Visual-inertial odometry is a fundamental technology exploited by autonomous vehicles and mobile robots to determine their position in unknown environments. Indeed, many techniques have been proposed for monocular and stereo cameras. State-of-the-art autonomous vehicles, however, are equipped with multiple cameras covering the entire vehicle environment. We present here MIxture of eXperts Odometry (MIXO), a data-driven, machine learning-based technique that loosely combines odometry outputs from multiple cameras to obtain a more accurate and robust global estimate. It stems from the intuition that each camera provides an optimal vantage point in specific driving scenarios. In MIXO, each camera (or expert) is individually processed by a state-of-the-art visual odometry algorithm (e.g., ORB-SLAM2). Then, the odometry estimates are mixed by a gating network, which selects the locally optimal experts in the current operational conditions and weights their contributions accordingly. MIXO is a lightweight module that can be easily implemented on top of any visual odometry algorithm. Experimental results on real-life data from autonomous vehicles show that MIXO achieves more robust and accurate results than any single camera, reducing the absolute rotational and translation error by 38% and 15%, respectively.**

***Index Terms*—Intelligent Transportation Systems, Visual-Inertial odometry, Mixture of Experts, Visual Odometry, Autonomous Vehicles**

I. Introduction

AUTONOMOUS vehicles and mobile robots need to be able to sense and map their surroundings in real-time in order to safely navigate unknown environments, while detecting and avoiding obstacles. To this aim, visual information provided by low-cost, consumer-grade sensors, such as cameras and inertial sensors, can be exploited [1], [2], [3]. The present research focuses on the fundamental odometry task, in which the agent's trajectory is estimated based on measurements recorded by the agent itself.

Inertial odometry, e.g., based on wheel revolutions or accelerometers, suffers from exponentially accumulating errors [4], [5]. Visual Odometry (VO), alone or in combination with inertial measurements, is a robust alternative as it is insensitive to uncertainties due to the terrain or sensor noise [6]. VO solves the inverse problem of estimating the agent's trajectory from the apparent motion of still elements (e.g., road, buildings) in the surrounding environment. Many algorithms have been proposed to solve the VO problem for monocular [7], [8], [9], [10], stereo [7] and RGB-D cameras [7], [11].

Modern autonomous vehicles are typically equipped with several cameras (six or more) to cover the complete vehicle's surroundings [12] with minimally overlapping fields of view. Yet, in the majority of VO literature, a monocular or at most stereo camera placed in front of the vehicle is considered. Few algorithms have been proposed to integrate information from different complementary cameras [13], [14], [15], [16], [17], [18].

Our contribution stems from the intuition, which we support with ample experimental data from the nuScenes dataset [12], that in typical urban scenarios different cameras do not provide equally reliable information. Instead, each camera offers a unique vantage point and its accuracy varies depending on the specific driving maneuvers (e.g., right or left turn) and surrounding scene. For this reason, we propose a novel way to dynamically weight the contribution of each camera using a data-driven approach [19]. The Mixture of eXpert Odometry (MIXO) technique obtains a loosely coupled joint estimate for the vehicle trajectory, building upon individual odometry measurements that are less computationally demanding than solving a high-dimensional joint optimization problem. The propose method can also be extended to fuse camera information with other sensors (e.g., inertial).

The rest of the paper is organized as follows. Relevant literature is discussed in Section II. The main principles of MIXO are introduced in Section III, while Section IV illustrates the experimental setup in which it was evaluated. Experimental results are presented and discussed, along with future works, in Sections V and VII, respectively.

II. Related Work

A. Visual Odometry

VO techniques can be classified based on the type of camera (e.g., monocular, stereo or RGB-D), fusion with other sensors (e.g., visual-inertial odometry), and how keypoint information is extracted (direct, feature-based, hybrid). Among the latter, feature-based approaches provide the best trade-off between accuracy and computational requirements [1], [2]; they compute the camera motion by matching feature descriptors across consecutive frames.

For stereo and RGB-D cameras, depth information can be acquired or reconstructed by triangulation; thus, it is possible to compute the ego-vehicle motion with known real-world scale. In monocular cameras, only the pose of the ego-vehicle can be directly estimated. To overcome this problem, other assumptions about vehicle movement, environment, or other sensors are needed. In Visual-Inertial Odometry, classical inertial navigation systems are fused with VO to estimate the trajectory scale and overcome issues with poor lighting, feature-less scenes and accelerometer noise. Alternatively,

Lia Morra, Andrea Biondo and Fabrizio Lamberti are with the Department of Control and Computer Engineering, Politecnico di Torino, Turin, Italy. Nicola Poerio is with Stellantis SpA, Turin, Italy.

Manuscript received April 19, 2021; revised August 16, 2021.

scale information can be calculated from depth measured by LIDAR [20] or estimated by an ad-hoc neural network [8].

In this work, we base our baseline VO algorithm on the well-known ORB-SLAM2 technique [7], which has competitive performance on the KITTI benchmark [21], can be run in real time and an open source implementation is available. ORB-SLAM2 is compatible with monocular, stereo, and RGB-D cameras. To estimate the scale, we relied on loose integration with inertial odometry.

B. Multi-Camera Visual Odometry

Abrupt camera movements, unfavorable lighting conditions or outliers in feature matching are only some of reasons why a single camera setup could fail to provide an accurate estimate of the ego-vehicle pose. Experiments performed in a variety of settings, including indoor scenarios and unmanned aerial vehicles, show increased robustness with respect to single camera VO [17], [13], [14], [22], [15].

Multi-camera visual odometry algorithms can be classified in *loosely* and *tightly coupled* approaches [1]. In loosely coupled approaches, the position of the ego vehicle is estimated independently of each sensor, and the estimated data is later fused into a single estimated trajectory [23], [24].

Tighly-coupled approaches, on the other hand, take directly into account the multi-camera setup when solving the motion estimation problem [13], [14], [15], [25], [26]. Although the proposed methods differ with respect to how feature information is retrieved, all integrate the keypoints obtained from different cameras in the same cost function. One possible approach minimizes the sum of the reprojection errors of all detected keypoints, weighted by the estimated noise of each point and the standard deviation of all reprojection errors [14]. Performance-wise, tighly-coupled approaches are usually deemed to yield better results compared to loosely-coupled techniques [1]; the latter, on the other hand, limit the computational load by employing a fixed-dimension state space. For this reason, techniques for keypoint selection have been proposed to limit the computational burden associated with solving the joint estimation problem [25].

A common problem of integrating multiple sensors, characterized by different uncertainties and errors, is how to weight and combine their contributions, especially in a loosely-coupled fashion. We are particularly interested in addressing challenges associated with autonomous vehicles equipped with multiple monocular cameras. In this context, the quality of the trajectory estimated from each camera depends on the ego-vehicle position with respect to the road structure (e.g., how fast does the scene change, whether the scene has sufficient texture), on the vehicle maneuvre, on the number of moving targets in the field of view, etc. Our goal is to investigate whether data-driven techniques are capable of dynamically weighting the contribution of each camera in order to leverage a multi-camera setup while at the same time minimizing the computational overhead.

C. Mixture of Experts

Mixtures of Experts are formulated as a parameterized ensembling technique, first proposed by Jacobs et al. [27]. In a MoE, several specialized models (or *experts*) approximate the target function in a subset of the input or output space. Their outputs are combined using a parametric gating function or gate, often implemented as a neural network, that is trained to dynamically select the best possible expert combination for each input [19], [28]. Given their flexibility and expressive power, MoEs have been widely adopted in both classical machine learning and deep learning techniques. MoE applications are vast and range from generally improving classification performance, to sensor fusion and uncertainty estimation. In the field of computer vision, MoE techniques have been applied to image classification [29], [30], object detection [31], segmentation [32], [33] and human pose estimation [34]. To the best of our knowledge, this is the first attempt to apply this framework in the VO task.

Experts may be *implicitly* specialized if they operate on different modalities or views [31], [32], or have different architectures or inductive biases [33]. This is a common scenario in sensor fusion, since different experts are assigned to different modalities and therefore may be optimal under specific operating conditions. In person detection, a MOE-based architecture was used to automatically switch between RGB and depth sensors depending on visibility and illumination conditions [31]. The main advantage is that the MoE is trained to perform the switch without having to explicitly or manually define operating conditions for each sensor.

In other settings, specialization is *explicitly* induced by partitioning the input space, e.g., via random sampling or according to a predefined strategy [19], [33], [34]. For instance, Pavlitskaya et al. designed a MoE network for semantic segmentation in which each expert specializes on a specific driving scenario, assigned either manually (urban or highway) or automatically (based on the sky-to-drivable ratio) [33]. In the case of MIXO, the experts operate on both different modalities, as well as on different partitions of the input space, implicitly induced by the different orientation of each camera.

Another important consideration when designing a MoE technique is how the gating function is integrated with the experts. When both the function and the experts are based on deep learning, it is reasonable to assume that they share the same input embedding and are co-trained in an end-to-end fashion [29], [30], [31], [33], [34]. However, the MoE framework does not require that the individual experts are neural networks and can be applied also when the experts are non-differentiable, as in our case. For this reason, in MIXO an ad-hoc optimization problem was defined to train the gating function.

III. MIXO: Mixture-of-Expert Visual Odometry

The proposed technique targets a multi-camera setup in which each monocular camera covers a specific direction (e.g., frontal, lateral, etc.). The objective of MIXO is to combine individual trajectory estimates, obtained independently for each camera, weighting the contribution of each camera based on the current vehicle state, trajectory and surrounding. Before delving into the design of MIXO in Section III-C, the single-camera VO algorithm adopted as reference is summarized in Section III-B.

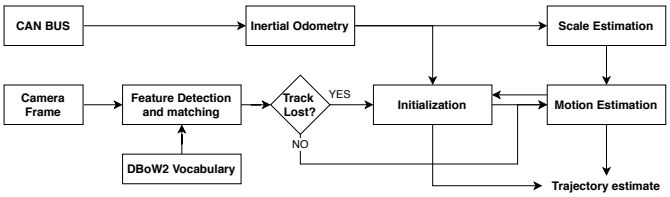


Fig. 1. Single-camera algorithm architecture.

A. Notation

Let us consider a set of N sensors (e.g., cameras, IMU, etc.), each associated with a set of k consecutive measurements (e.g., video frames in the case of cameras). We assume that, for a given sensor j, an odometry algorithm is capable to estimate the relative pose of the ego-vehicle at frame Y_k^j, with respect to frame $k-1$. The relative pose can be decomposed as $Y_k^j = \left[{t'}_k^j, s_k^j, \psi_k^j\right]$, where ψ_k^j is the yaw angle, and t_k^j is the translational term provided by the motion estimation algorithm ${t'}_k^j$ and re-scaled according to the scale factor s_k. The multi-camera, multi-sensor odometry problem is framed, in a loosely-coupled setting, as the optimization problem of finding the optimal global relative pose Y_k given the individual relative poses $Y_k^j, j = 1, ..., N$.

B. Single-Camera Odometry

The proposed single-camera implementation, illustrated in Fig. 1, is based on the open-source ORB-SLAM2[1] implementation [7], adapted to the specific characteristics of the nuScenes monocular camera setting. The main components are briefly summarized here.

At each incoming frame, rotation and translation of each camera are estimated by matching features across consecutive frames. *Feature detection and matching* are based on the ORB descriptor [35]. ORB detects salient keypoints in the image by thresholding the output of the FAST corner detector: to maximize the number of keypoints, the image is divided into a grid, and the threshold is gradually decreased from an initial value *iniThFAST*, until a minimum number of corners (i.e., keypoints or features) is detected, or a minimum threshold *minThFAST* is reached. The orientation and ORB descriptor are then computed on the retained FAST corners. A pyramid of features calculated at different scales is exploited to achieve scale invariance: *nLevels* is the total number of levels in the feature pyramid, and *scaleFactor* is the resizing between adjacent levels of the scale pyramid. Feature matching is solved by a brute-force approach, with mutual consistency check. The search is further constrained using a Bags of Binary Words (DBoW2) representation, i.e., only features that belong to the same node in the vocabulary tree are matched [36].

In monocular visual odometry, only rotation and translation parameters are obtained. We thus perform *scale estimation* externally based on an *Inertial Odometry* model, which is described in Section IV-B3. Inertial Odometry was also used to calculate the trajectory during the ORB-SLAM2 *initialization* phase.

[1]available at https://github.com/raulmur/ORB_SLAM2

The final step is *motion estimation* which is based on a planar 2D-to-2D model, followed by local bundle adjustment to minimize the re-projection error of the selected features and prevent scale drift.

Given the output of the odometry algorithm $Y_k = [{t'}_k, s_k, \psi_k]$ for a given frame k, the successive poses are concatenated according to

$$P_k = P_{k-1} + \begin{bmatrix} cos(\psi_k) & -sin(\psi_k) & 0 \\ sin(\psi_k) & cos(\psi_k) & 0 \\ 0 & 0 & 1 \end{bmatrix} t_k \tag{1}$$

where ψ_k is the yaw angle, and t_k is the translational term provided by the motion estimation algorithm ${t'}_k$ and re-scaled according to the scale factor s_k.

C. Mixture of Experts for Multi-Camera Visual Odometry

Let us consider a set of N sensors (e.g., cameras, IMU, etc.), each associated to a set of relative pose estimates $Y_k^j = \left[{t'}_k^j, s_k^j, \psi_k^j\right]$, where $k = 1...M$ is the time frame and $j = 1...N$ is the sensor. Our goal is to calculate a global trajectory estimate $Y_k = [{t'}_k, s_k, \psi_k]$ as a weighted linear combination of individual trajectories Y_k^j, and to estimate the optimal weights $\mathbf{w}_k^* = f(\mathbf{x}_k)$, where $\mathbf{x}_k$ is a set of measurements representing the current state of the vehicle and the maneuvering maneuver. The intuition behind this problem formulation is that, at each time point, one or more cameras provide a locally optimal trajectory estimate: intuitively, the frontal camera is expected to provide a better estimate when the vehicle is driving straight, whereas lateral cameras offer a better vantage point during fast turns.

To support the intuition behind MIXO the nuScenes dataset was analyzed to determine, at each time point, which camera allows to estimate the incremental yaw rates with lowest error. The following optimization problem was solved through an exhaustive search:

$$\arg\min_j ||(\psi_k^{GT} - \psi_{k-1}^{GT}) - (\psi_k^j - \psi_{k-1}^j)||_2 \tag{2}$$

where ψ_k^{GT} is the yaw ground truth (GT) at instant k. The frontal camera was found to be locally optimal only in 39% of the timepoints, whereas the other cameras were optimal on average 20% of the time.

Moving from these considerations, multi-camera VO can be cast as a typical Mixture of Experts (MoE) problem [19]. The MoE technique is based on the divide-and-conquer principle: a set of so-called "experts", which are capable of solving a problem locally but not globally, are mixed through a gating function which selects the best expert(s) in the current operational condition and weights their contributions. In MIXO, the experts represents the cameras, or better the instances of the single-camera odometry algorithm that processes each camera stream, and any other sensor that provides a trajectory estimate. The proposed technique can thus, in principle, be applied to any monocular odometry algorithm.

The architecture of the proposed system is shown in Fig. 2. For simplicity, we consider in the following only the calculation of the incremental yaw rate $\Delta\psi_k$ as a combination

of the incremental yaw rates estimated by each expert $\Delta\psi^j$, where $j \in \{1, .., N\}$ and N is the number of experts. Relative (i.e., incremental) instead of absolute poses are fused at each time steps. In our single-camera odometry, scale is estimated externally the same inertial odometry model for all cameras, and hence the gating function learns to predict the optimal yaw rate combination.

MIXO predicts the optimal weighting scheme taking as input a vector $\mathbf{x}$ that comprises the yaw rate at the previous time instant $\Delta\psi_{k-1}$, the longitudinal velocity v_{k-1}, and the number of matched features for each camera $m_{k,j}$. The yaw rate and velocity encode the current driving scenario (they distinguish, for instance, a rectilinear trajectory from a sharp turn), whereas $m_{k,j}$ are indicative of the scene characteristics. The gating function is a neural network which outputs the weights assigned to each expert [19]. The final incremental yaw is then computed as the weighted linear combination of the individual incremental yaws $\Delta\psi_k^j$:

$$\Delta\psi_k(\mathbf{x}_k) = \sum_{j=1}^{N} h(w_j(\mathbf{x}_k))\Delta\psi_k^j \tag{3}$$

where k is the current time point, $\mathbf{x}_k$ is the input vector at time k, $w_j(\mathbf{x}_k)$ is the weight computed by the gating network for the j^{th} expert and $h(\cdot)$ is the softmax function, which ensures that the sum of the weights is equal to 1. The final yaw is computed as:

$$\psi_k = \psi_{k-1} + \Delta\psi_k(\mathbf{x}_k) \tag{4}$$

The model parameters are obtained by minimizing the mean square error (MSE) of the estimated yaw rate:

$$\mathbf{w}_k^* = \arg\min_{\mathbf{w_k}} \frac{1}{M} \sum_{k=0}^{M} \left(\psi_k^{GT} - \sum_{j=1}^{N} h(w_j(\mathbf{x}_k))\psi_k^j \right)^2 \tag{5}$$

where M is the total number of time points and N is the total number of experts.

We experimented also with other gating functions based on a linear model, but the results were not satisfactory denoted as LINEAR in Section IV-D, showing no improvement over the single-camera estimate.

IV. Experimental Settings

In this section, the experimental setup is introduced. The dataset is detailed in Section IV-A. Implementation details of the single-camera multi-camera VO algorithms are detailed in Sections IV-B and IV-C, respectively. MIXO is compared against several baselines, described in Section IV-D, using the evaluation protocol detailed in Section IV-E.

A. Dataset

All experiments were performed on the nuScenes dataset [12]. This dataset includes a total of 15h of driving data collected in Boston and Singapore, divided in 1000 scenes of 20s length. Compared to KITTI, nuScenes is larger and contains more challenging acquisition scenarios (such as rainy days, parking lots, or suburban areas). The vehicles were equipped with 6 RGB cameras sampled at 12Hz, GPS and

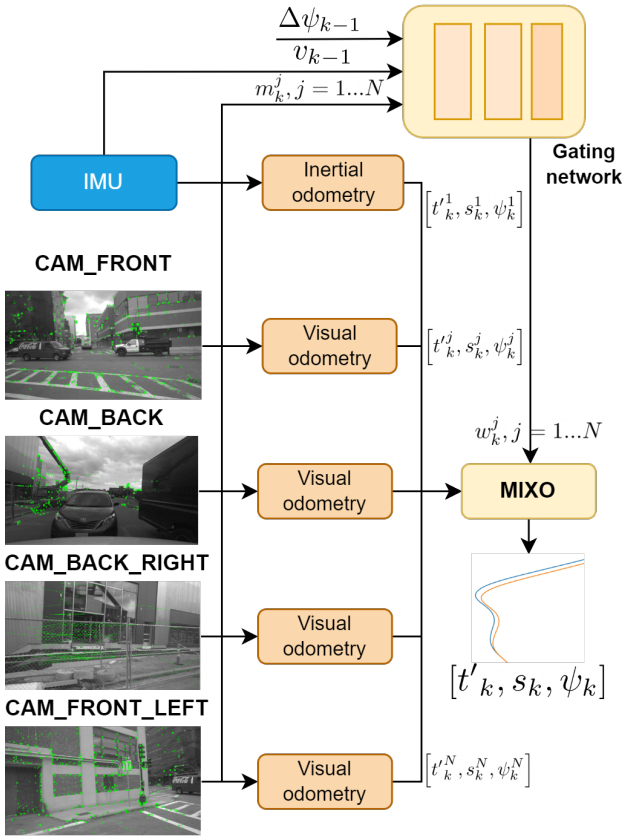


Fig. 2. MIXO architecture. Keypoint detection and matching is performed on each individual camera stream. Yaw rate estimates provided by each expert (camera) are weighted by a gating network, which takes as input the yaw rate and velocity at the previous frames, as well as the number of matched keypoints in the current frame. The gating network is composed by two fully connected layers, with dropout, followed by a softmax layer.

IMU sensors. The CAN bus data expansion, which provides raw sensor measurement from the IMU sensor, was exploited to calculate the inertial odometry and retrieve the longitudinal velocity (the velocity in m/s, expressed in the ego vehicle frame, is sampled at 50Hz). The ego-pose in the nuScenes annotations, sampled at 2Hz, was used as the ground truth. Details of sensor setup (including reference frames and FOV for each camera), CAN bus data expansion, and ego-pose reference standard are provided in [12]. Night scenes were excluded since ORB-SLAM2 could not detect enough features. The final dataset includes 736 scenes from the nuScenes training set, for a total travelled distance of 176km.

B. Single-Camera Odometry Implementation

1) Feature detection and matching: The parameters were manually selected on a subset of the nuScenes training set to balance the sensitivity and specificity of the feature matching. Compared to the KITTI [21] dataset, the total number of FAST corners to be extracted was increased from 2000 to 8000, and *nLevels* from 8 to 19, to account for the higher image resolution. The other parameters were left unchanged (*scaleFactor*=1.2, *iniThFAST*=20, *minThFAST*=7). Loop closure and visual place recognition were deactivated. ORB-SLAM2 implementation provides a basic vocabulary

extracted from the Bovisa dataset [36], [7]. For lateral cameras (CAM_BACK_RIGHT and CAM_FRONT_LEFT), a new vocabulary was trained using a randomly chosen 20% of the overall image dataset.

2) Initialization: Initialization takes place whenever the VO system has just started or when a track loss occurs, that is the matched features are less than 50. It could take several seconds depending on the scene settings; since nuScenes includes only scenes that are 20s long, not being able to correctly estimate the position in the first seconds deeply affects the estimated performance. For this reason, inertial odometry was used to provide a good initial estimate for the trajectory.

3) Inertial odometry: A classical inertial odometry was implemented using a discrete Kalman filter, considering a kinematic 2D model for the vehicle's pose in which the state of the system is defined as $X_k = [s_k, v_k, \psi_k, \Delta\psi_k]^T$, with s_k the overall traveled distance and ψ_k the yaw angle, and given the longitudinal velocity v_k (available in nuScenes via the CAN bus data extension) and the yaw rate $\Delta\psi_k$ as input.

Since the initial pose is retrieved from the ground truth, X_0 is considered to be perfectly known and therefore its related covariance matrix P is a 4×4 null matrix. Moreover, it is also assumed that v and $\Delta\psi$ are independent white Gaussian process. Given the characteristics on the error bound reported in nuScenes in the measurement of the speed and the yaw rate, we set the measurement noise matrix to:

$$R = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.01 \end{bmatrix} \quad . \tag{6}$$

The scale of the scene, defined as the total distance traveled on the longitudinal axis between two consecutive frames, is estimated by subtracting the distance traveled at time t_k from the distance traveled at time t_{k-1}.

C. Mixture of Experts

In the present implementation, MIXO combines five experts: four cameras (CAM_FRONT, CAM_BACK, CAM_BACK_RIGHT, and CAM_FRONT_LEFT) and the inertial odometry. In order to lower the computational requirements, while balancing over- and under-fitting, the model used was a simple feedforward neural network with one input layer, two hidden dense layers, each followed by a dropout layer to prevent overfitting, and a final softmax activation. All input data were rescaled to zero mean and unitary standard deviation.

Results were calculated using 4-fold cross-validation. The best values of the hyper-parameters were selected through grid search on a randomly selected validation split. The Adam optimizer was used with exponential decay learning rate schedule $\eta(t) = \eta_0 \alpha^{t/t_0}$, initial learning rate η_0 and decay rate α. The hyper-parameters to optimize are the number of units in the two hidder layers N_1 and $N_2 \in \{8, 16, 32, 64, 128\}$, the dropout rates d_1 and $d_2 \in \{0, 0.1, 0.2, 0.4\}$, the initial learning rate $\eta_0 \in \{0.001, 0.005, 0.01, 0.02, 0.05\}$ and the decay rate $\alpha \in \{0.001, 0.01, 0.05, 0.1, 0.5\}$. The optimal parameters were $N_1 = N_2 = 16$, $d_1 = d_2 = 0.1$, $\eta_0 = 0.02$ and $\alpha = 0.05$.

D. Baselines

The proposed MoE model is based on a few assumptions, i.e., that each expert is locally, but not globally, optimal, and that the optimal camera depends on the current driving scenario. To verify the validity of these assumptions, MIXO was compared against three baselines:

- a weighting scheme that selects the camera with the highest number of matched features (MATCH);
- a constant weighting scheme (LINEAR);
- selection of the optimal camera (i.e., the one that yields the yaw rate closest to the GT) at each time step (MULTICAMERA GT).

In the constant weighting scheme, the incremental yaw is calculated as the linear combination of the individual incremental yaws, as in MIXO , but each camera is assigned a constant weight w_k by minimizing the RMSE. The last baseline is not a proper weighting scheme, as the GT is not available at inference time; it is calculated to estimate the best performance achievable, given the current VO algorithm, by selecting only the best camera at each given time frame.

E. Performance Evaluation

The metrics used to evaluate our model are the absolute trajectory error (ATE) and the relative trajectory error (RTE), calculated separately for the rotation and translation, and the average number of lost tracks per scene. A detailed overview of the two metrics, along with their advantages and drawbacks, is provided in [37]. The absolute trajectory error yields a single and hence was selected as the primary metric for ease of comparison. Performance metrics were calculated using the `rpg_trajectory_ evaluation` software package [37].

Briefly, the ATE is calculated aligning the GT and estimated trajectories (in our case, this is not necessary as the first pose is taken from the ground truth), and then calculating the root MSE (RMSE), for both translation (t_{abs}) and rotation (r_{abs}).

The RTE measures the relative relations between the poses at different times. The path is parameterized along the traveled distance and the trajectory is then divided in K subsets. Each trajectory subset is then aligned with the GT as in the ATE calculation; then each sample is paired with its consecutive one and for each pair d_k the relative error is calculated.

This equation yields scalar values for both the angular and position error for each of the K subset trajectories the path has been divided in.

V. Results

In this section, the performance of MIXO is compared against single-camera VO. The RTEs for all experiments, either single- or multi-camera, are summarized in Table I.

A. Single-Camera Odometry

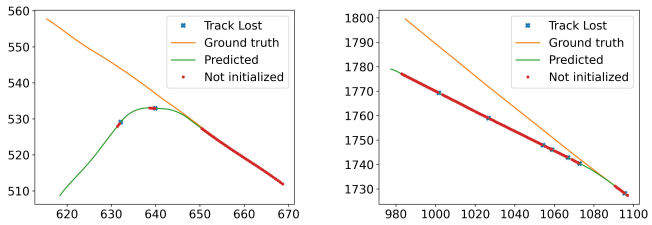
When taken individually, the frontal camera achieves the best performance with t_{abs}=1.68% and r_{abs}=0.029 deg/m. Without initializing the trajectories through inertial odometry, the ATE dramatically increases to t_{abs}=7.37%, and r_{abs}=0.238 deg/m. On the KITTI dataset, ORB-SLAM2 achieves a

TABLE I

ABSOLUTE (TRANSLATION AND ROTATION) TRAJECTORY ERROR FOR SINGLE AND MULTI-CAMERA ALGORITHMS. TRANSLATION AND ROTATION ERRORS ARE GLOBAL STATISTICS COMPUTED OVER THE ENTIRE TRAVELLED DISTANCE. THE AVERAGE NUMBER OF LOST TRACK IS COMPUTED PER SCENE (MEAN ± STANDARD DEVIATION). ALL EXPERIMENTS INCLUDE INITIALIZATION WITH INERTIAL ODOMETRY AND THE ORIGINAL DBOW VOCABULARY PROVIDED WITH ORB-SLAM2, UNLESS OTHERWISE STATED. BEST PERFORMING SOLUTIONS (EXCLUDING MULTICAMERA GT) ARE IN BOLD.

Algorithm	**Transl. [%]**	**Rot. [deg/m]**	**Avg. track lost**
Single-camera odometry			
CAM_FRONT [w/o inertial initialization]	7.37%	0.238	0.89 ± 2.58
CAM_FRONT	**1.68%**	**0.029**	**0.76 ± 1.93**
CAM_BACK	3.85%	0.077	**0.74** ± 3.14
CAM_BACK_RIGHT	6.75%	0.241	5.83 ± 6.20
CAM_BACK_RIGHT [w/ retrained vocab.]	5.57%	0.146	4.43 ± 4.89
CAM_FRONT_LEFT	8.93%	0.232	7.00 ± 5.58
CAM_FRONT_LEFT [w/ retrained vocab.]	7.59%	0.204	4.68 ± 5.58
Multi-Camera Odometry			
HIGHEST MATCH	2.18%	0.072	0.00 ± 0.00
LINEAR	1.57%	0.026	0.00 ± 0.00
MIXO	**1.44%**	**0.019**	**0.00 ± 0.00**
MULTICAMERA GT	1.46%	0.012	0.00 ± 0.00

Fig. 3. Example of a scene in which the CAM_BACK_RIGHT fails due to the high number of moving objects (cars) present in the field of view.



(a) Scene-0533 (b) Scene-0892

Fig. 4. Examples of incorrect trajectories (green) estimated using by the CAM_BACK_RIGHT camera, compared with the ground truth (yellow). Red marks indicate frames in which ORB-SLAM2 is initializing, blue marks frames in which the track is lost.

t_{abs}=1.15% and r_{abs}=0.027 deg/m on the stereo frontal camera [7]. However, trajectories in KITTI are much longer than in the nuScenes dataset, which is divided in 20s-long sequences. The impact of the initialization phase is thus much higher in nuScenes, and results cannot be directly compared. Empirically, we also observed a higher number of low-textured scenes in nuScenes, which causes ORB-SLAM2 to fail to initialize or to lose track, with an average of 0.762 lost tracks per scene. Additionally, in KITTI a stereo camera is available, which drastically reduces scale drift problems. Nonetheless, it is worth noticing that the ORB-SLAM2 vocabulary generalizes well to nuScenes.

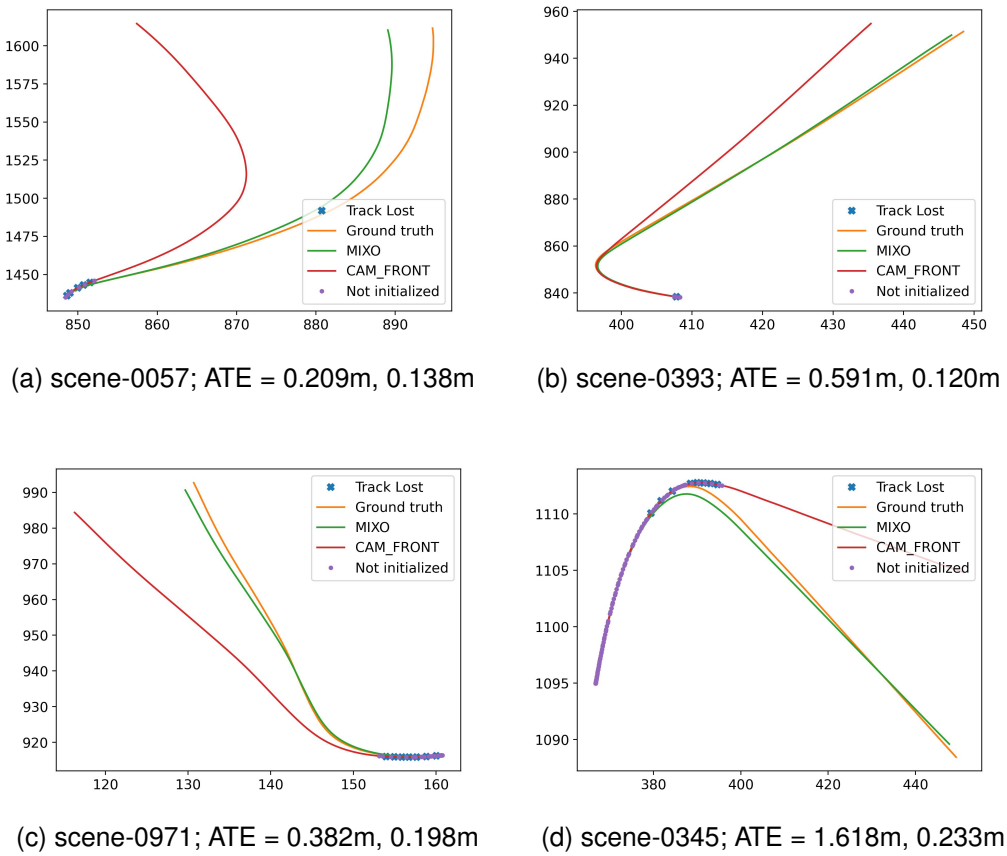
As evident from Table I, the performance of CAM_BACK is slightly inferior to that of CAM_FRONT. In a few scenes, the road is very busy and a large number of features are detected and matched on moving vehicles. These outliers are generally removed using RANSAC, which however fails when the number of false positive matches is too high, as illustrated by the exampl in Fig. 3.

The lateral cameras provide worse results with respect to the frontal and back cameras. Retraining the DBoW2 vocabulary improves the performance by roughly 20%. For instance, for CAM_BACK_RIGHT t_{abs} improves from 6.75% to 5.57% and r_{abs} from 0.241 to 0.146 deg/m. The results are still much worse than the frontal and back cameras. We attribute this result to the fact that, when the car is going straight (that is, for most of the time), the apparent motion of the lateral camera is much faster than that of the frontal camera, leading to incorrect feature mapping. Some illustrative examples are provided in Fig. 4. However, when the vehicle is turning, the lateral camera perceives a slower motion thus yielding better results. The complementary role of each camera can be exploited by a multi-camera VO.

B. Multi-camera odometry

As illustrated in Table I, MIXO outperforms any single-camera odometry, including the frontal one, achieving t_{abs}=1.44% (-15%) and r_{abs}=0.019 deg/m (-38%). The rotational error is reduced by a much larger factor, which is easily explained by taking into account that the translation and scale is estimated externally, using the same inertial measurements for each camera. Another advantage of MIXO is that it never loses track of the ego-vehicle, as it needs only one camera to detect a sufficient number of features. Fig. 5 illustrates some challenging scenarios, including high curvature turns, in which the trajectories estimated by MIXO are clearly superior to those estimated by the single-camera VO (CAM_FRONT).

The distribution of the ATE per scene is further compared through the boxplots in Fig. 6. While the frontal camera alone struggles in high challenging, low-textured scenarios achieving



(a) scene-0057; ATE = 0.209m, 0.138m (b) scene-0393; ATE = 0.591m, 0.120m

(c) scene-0971; ATE = 0.382m, 0.198m (d) scene-0345; ATE = 1.618m, 0.233m

Fig. 5. Examples of ego-vehicle trajectories: computed by MIXO (green) and single-camera VO (CAM_FRONT, red), compared with ground truth (yellow). Purple marks indicate frames in which ORB-SLAM2 is initializing, blue marks lost tracks; in both cases, inertial odometry is used.

errors higher than 0.5m on 29 out of 736 scenes, MIXO shows a more robust behavior with a smaller error range.

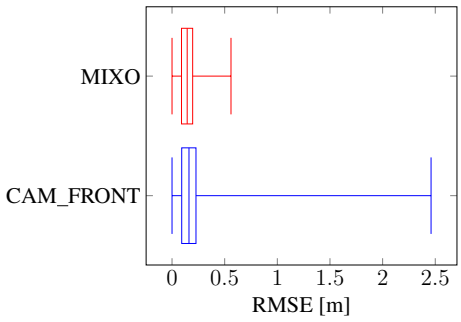


Fig. 6. Distribution of ATE (RMSEs) per scene for CAM_FRONT VO (blue) and MIXO (red).

A comparison with alternative, and simpler, weighting schemes, confirms the validity of MIXO's assumption. A linear combination of the individual yaw rates marginally improves over the single-camera estimate, probably due to the integration of the inertial odometry. The weights obtained by minimizing the RMSE on the training set were $\mathbf{w} = [0.851, 0.020, 0.031, 0.013, 0.1222]$, referring, respectively, to the FRONT, FRONT_LEFT, BACK, BACK_RIGHT camera, and the inertial odometry. Selecting the camera with the highest number of matched features provides even worse results than using the frontal camera alone, mostly due to the confounding effect of scenes that yield a higher number of false positives matches. On the other hand, MIXO performance is close to that achieved by an oracle which always selects the camera that provides the best incremental yaw rate, denoted as MULTICAMERA GT in Table I. In Table II, t_{abs} and r_{abs} are separately reported for each split as well as averaged over the entire dataset to illustrate the variance due to different training/validation sets.

TABLE II
MIXO RESULTS FOR EACH SPLIT AND THE OVERALL DATASET.

Transl. [%]	**Rot. [deg/m]**	**Dataset**
1.44%	0.018	Scenes 1-184
1.43%	0.016	Scenes 185-368
1.44%	0.021	Scenes 368-552
1.46%	0.022	Scenes 553-736
1.44%	0.019	Overall

Finally, the distribution of the RTE as a function of the traveled distance, from 10 to 200m, is illustrated in Fig. 7. One of the main limitations of our experiments is the short duration of the scenes (20s). As a result, the initial RTEs are high due to the initialization process, which relies only on the inertial odometry; afterwards, the error starts to converge,

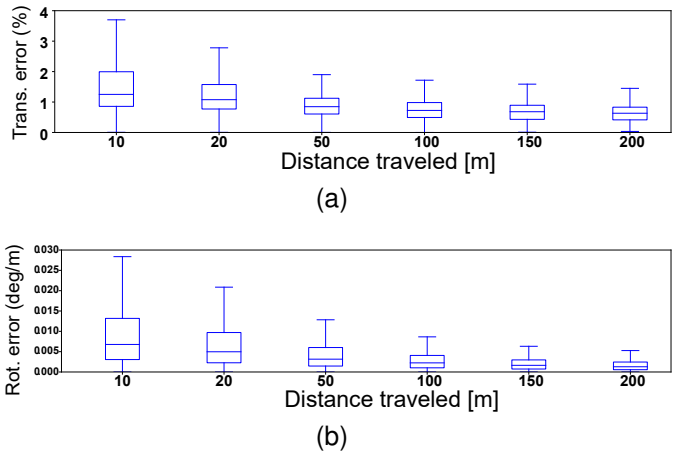


Fig. 7. Relative error computed at different sub-trajectory lengths: translational (a) and rotational (b) errors.

with a mean final value of 0.7% for translation and 0.002 deg/m for rotation. It is reasonable to assume that since the error converges at higher distances, MIXO should be able to provide a good pose estimate also in longer sequences.

VI. Discussion

There are two main challenges associated with the adoption of data-driven, multi-camera VOs. The first are the computational requirements deriving from the need to simultaneously process multiple video streams, e.g., between four and six in typical automotive settings.

MIXO includes a very lightweight gating network with the goal of introducing minimal overhead on top of the single-camera VO algorithm. Hence, the main processing load depends on the characteristics of the input streams (framerate, resolution) and the computational requirements of the single-camera VO algorithm. For reference, nuScenes sensor setup comprises 6x Basler acA1600-60gc operating at 12Hz with 1600×1200 resolution, whereas KITTI (another popular dataset for VO research) was acquired at 10Hz using two stereo cameras acquired with 1226×370 resolution.

Our experiments were based on ORB-SLAM2, a computationally efficient method that does not require GPU optimization. However, execution time is strictly coupled to a specific computing platform and implementation. Depending on the input size, the average execution time on a 4-8 core desktop CPU ranges between 10 ms and 80 ms/frame, considering our experiments as well as previous literature [38], [39]. On automotive embedded platforms, such as NVIDIA Tegra TX1, execution times up to 190 ms/frame have been reported [39]. Specialized implementations may reduce the execution time on embedded platforms to 62 ms/frame with CPU pipelining and 30 ms/frame with additional GPU support [40]. It should be noted, however, that most papers report experiments on lower-resolution frames from the KITTI dataset. Even under these more favorable conditions, with the reported times at most two cameras could be processed within real-time requirements (83.3 ms/frame). To achieve real-time execution on current automotive embedded systems, further reduction in execution time is needed.

Another downside of MIXO, and data-driven approaches in general, is that it is trained in a supervised manner. Hence, it requires a large dataset with known reference standard, and may not generalize well to different scenarios, unseen during training. As shown in Table II, despite a certain variance between the performance observed on different splits, all networks outperform the single-camera solution. However, even though nuScenes is a relatively large dataset, all scenes were acquired with the same sensor setup. Hence, there is no guarantee that the gating function trained on nuScenes will generalize to different sensor setups.

One of the advantages of MIXO is that it implicitly learns to downweight cameras that, based on the current driving maneuver, have a high probability to contain low-textured scenes or moving targets. Other techniques have tried to explicitly avoid integrating moving objects in the motion estimation by exploiting semantic segmentation [41], which however is a computationally expensive step at both training and inference time. Further experiments are needed to determine whether the implicit association learnt in an automotive scenario can be transferred, and to what extent, to other types of autonomous systems (e.g., drones or indoor robots).

VII. Conclusions and Future Works

This paper introduces MIXO, a multi-camera loosely-coupled visual-inertial odometry technique that exploits the complementary information provided by different cameras and sensors by framing the odometry task as a MoE problem. Each expert consists of a single-camera odometry algorithm – in principle, any algorithm can be used for this purpose – applied to one of the cameras. It is not required that the fields of view of adjacent cameras overlap.

Experiments on the nuScenes dataset, integrating four single-camera monocular odometry outputs (FRONT, FRONT_LEFT, BACK, BACK_RIGHT), demonstrate the feasibility of loosely coupling multiple trajectories estimates using a data-driven approach.

The proposed methodology could be expanded in several ways. The scale estimate could be improved by leveraging different monocular odometry algorithms that can integrate depth estimation using LiDAR or monocular depth estimation networks [8], [20]. Semantic segmentation could be also exploited to avoid integrating moving objects in the motion estimation [41].

Multi-camera VO comes at the expense of increased computational requirements, making it more difficult to achieve real-time efficiency. This consideration applies to both loosely and tightly coupled approaches, although MIXO avoids the additional cost of solving a complex joint motion estimation problem. Hence, comparison with tightly coupled estimation, both from an accuracy and computational perspective, should be explored in future work.

In a loosely coupled approach, like MIXO computational efficiency can be further improved in several ways. One possible direction is to compute the full motion estimate only for the most relevant cameras. In the current formulation of the gating function, only the feature calculation and matching

steps are required to predict the weight associated with each expert. In future work, the computational burden could be further reduced by predicting the optimal camera(s) before feature extraction and matching, e.g., by using a dedicated convolutional neural network, and thus processing only the most useful camera(s).

Another interesting perspective is to integrate recent data-driven techniques for direct odometry estimation, which directly regress rotation and translation parameters from input images [38], [42]. Compared to feature-matching techniques, data-driven approaches can leverage the representation capabilities of deep neural networks to improve robustness under non-ideal conditions, as well as take advantage of the high parallelism of GPU optimization. However, the large training set required, as well as their lack of robustness to shifting operating conditions and sensors, has so far stifled their adoption. In a purely data-driven setting, MIXO could be expanded to integrate the gating function with the VO network, enabling end-to-end training of the ensemble, possibly exploiting attention mechanisms to further improve sensor fusion.

As a final remark, research in data-driven VO would benefit from large and diverse datasets with longer sequences, acquired with different camera installations, and in a wider range of urban and sub-urban contexts.

ACKNOWLEDGMENT

This work was funded by a grant by FCA-CRF SpA.

REFERENCES

[1] S. A. S. Mohamed, M.-H. Haghbayan, T. Westerlund *et al.*, "A survey on odometry for autonomous navigation systems," *IEEE access*, vol. 7, pp. 97 466–97 486, 2019.

[2] M. He, C. Zhu, Q. Huang *et al.*, "A review of monocular visual odometry," *The Visual Computer*, vol. 36, no. 5, pp. 1053–1065, 2020.

[3] W. Zou, D. Wu, S. Tian *et al.*, "End-to-end 6DoF pose estimation from monocular RGB images," *IEEE Transactions on Consumer Electronics*, vol. 67, no. 1, pp. 87–96, 2021.

[4] A. Solin, S. Cortes, E. Rahtu *et al.*, "Inertial odometry on handheld smartphones," in *2018 21st International Conference on Information Fusion (FUSION)*. IEEE, 2018, pp. 1–5.

[5] E. Choi and S. Chang, "A consumer tracking estimator for vehicles in GPS-free environments," *IEEE Transactions on Consumer Electronics*, vol. 63, no. 4, pp. 450–458, 2017.

[6] D. Scaramuzza and F. Fraundorfer, "Visual odometry [tutorial]," *IEEE Robotics Automation Magazine*, vol. 18, no. 4, pp. 80–92, 2011.

[7] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source slam system for monocular, stereo, and RGB-Dcameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[8] N. Yang, L. v. Stumberg, R. Wang *et al.*, "D3VO: Deep depth, deep pose and deep uncertainty for monocular visual odometry," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1281–1292.

[9] M. He, C. Zhu, Q. Huang *et al.*, "A review of monocular visual odometry," *The Visual Computer*, vol. 36, no. 5, pp. 1053–1065, 2020.

[10] T. Qin, P. Li, and S. Shen, "VINS-Mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[11] S.-H. Liu, C.-C. Hsu, W.-Y. Wang *et al.*, "Improved visual odometry system based on kinect RGB-D sensor," in *2017 IEEE 7th International Conference on Consumer Electronics-Berlin (ICCE-Berlin)*. IEEE, 2017, pp. 29–30.

[12] H. Caesar, V. Bankiti, A. H. Lang *et al.*, "nuScenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 621–11 631.

[13] B. Resch, J. Wei, and H. P. A. Lensch, "Real time direct visual odometry for flexible multi-camera rigs," in *Computer Vision – ACCV 2016*, ser. Lecture Notes in Computer Science, vol. 10114. Cham: Springer International Publishing, 2017, pp. 503–518.

[14] S. Yang, S. A. Scherer, X. Yi *et al.*, "Multi-camera visual slam for autonomous navigation of micro aerial vehicles," *Robotics and autonomous systems*, vol. 93, pp. 116–134, 2017.

[15] C. Forster, Z. Zhang, M. Gassner *et al.*, "SVO: Semidirect visual odometry for monocular and multicamera systems," *IEEE transactions on robotics*, vol. 33, no. 2, pp. 249–265, 2017.

[16] S. Houben, J. Quenzel, N. Krombach *et al.*, "Efficient multi-camera visual-inertial slam for micro aerial vehicles," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 1616–1622.

[17] R. Hadsell, B. Matei, G. Salgian *et al.*, "Complex terrain mapping with multi-camera visual odometry and realtime drift correction," in *2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization Transmission*. IEEE, 2012, pp. 493–500.

[18] T. Sattler, B. Leibe, and L. Kobbelt, "Efficient & effective prioritized matching for large-scale image-based localization," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 9, pp. 1744–1756, 2017.

[19] S. Masoudnia and R. Ebrahimpour, "Mixture of experts: a literature survey," *The Artificial intelligence review*, vol. 42, no. 2, pp. 275–293, 2012.

[20] J. Zhang and S. Singh, "Visual-lidar odometry and mapping: low-drift, robust, and fast," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, vol. 2015-, 2015, pp. 2174–2181.

[21] A. Geiger, P. Lenz, C. Stiller *et al.*, "Vision meets robotics: The KITTI dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[22] M. Geppert, P. Liu, Z. Cui *et al.*, "Efficient 2d-3d matching for multi-camera visual localization," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 5972–5978.

[23] S. Lynen, M. W. Achtelik, S. Weiss *et al.*, "A robust and modular multi-sensor fusion approach applied to mav navigation," in *2013 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2013, pp. 3923–3929.

[24] R. Mur-Artal and J. D. Tardós, "Fast relocalisation and loop closing in keyframe-based slam," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 846–853.

[25] L. Zhang, D. Wisth, M. Camurri *et al.*, "Balancing the budget: Feature selection and tracking for multi-camera visual-inertial odometry," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1182–1189, 2021.

[26] K. Eckenhoff, P. Geneva, and G. Huang, "Mimc-vins: A versatile and resilient multi-imu multi-camera visual-inertial navigation system," *IEEE Transactions on Robotics*, vol. 37, no. 5, pp. 1360–1380, 2021.

[27] R. A. Jacobs, M. I. Jordan, S. J. Nowlan *et al.*, "Adaptive mixtures of local experts," *Neural computation*, vol. 3, no. 1, pp. 79–87, 1991.

[28] S. E. Yuksel, J. N. Wilson, and P. D. Gader, "Twenty years of mixture of experts," *IEEE transactions on neural networks and learning systems*, vol. 23, no. 8, pp. 1177–1193, 2012.

[29] C. Riquelme, J. Puigcerver, B. Mustafa *et al.*, "Scaling vision with sparse mixture of experts," *Advances in Neural Information Processing Systems*, vol. 34, pp. 8583–8595, 2021.

[30] X. Wang, F. Yu, L. Dunlap *et al.*, "Deep mixture of experts via shallow embedding," in *Uncertainty in artificial intelligence*. PMLR, 2020, pp. 552–562.

[31] O. Mees, A. Eitel, and W. Burgard, "Choosing smartly: Adaptive multimodal fusion for object detection in changing environments," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 151–156.

[32] V. Aggarwal, K. Nagarajan, and K. C. Slatton, "Multiple-model multi-scale data fusion regulated by a mixture-of-experts network," in *IGARSS 2004. 2004 IEEE International Geoscience and Remote Sensing Symposium*, vol. 1. IEEE, 2004.

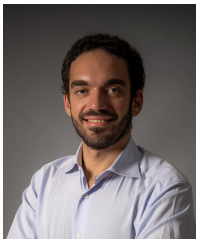
[33] S. Pavlitskaya, C. Hubschneider, and M. Weber, "Evaluating mixture-of-experts architectures for network aggregation," in *Deep Neural Networks and Data for Automated Driving*. Springer, Cham, 2022, pp. 315–333.

[34] E. Arnaud, A. Dapogny, and K. Bailly, "Tree-gated deep mixture-of-experts for pose-robust face alignment," *IEEE Transactions on Biometrics, Behavior, and Identity Science*, vol. 2, no. 2, pp. 122–132, 2019.

[35] E. Rublee, V. Rabaud, K. Konolige *et al.*, "ORB: An efficient alternative to SIFT or SURF," in *2011 International Conference on Computer Vision*. IEEE, 2011, pp. 2564–2571.

[36] D. Gálvez-López and J. D. Tardós, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, October 2012.
[37] Z. Zhang and D. Scaramuzza, "A tutorial on quantitative trajectory evaluation for visual (-inertial) odometry," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 7244–7251.
[38] G. Costante and M. Mancini, "Uncertainty estimation for data-driven visual odometry," *IEEE Transactions on Robotics*, vol. 36, no. 6, pp. 1738–1757, 2020.
[39] D.-D. Nguyen, A. Elouardi, S. A. R. Florez *et al.*, "Hoofr slam system: An embedded vision slam algorithm and its hardware-software mapping-based intelligent vehicles applications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 11, pp. 4103–4118, 2018.
[40] S. Aldegheri, N. Bombieri, D. D. Bloisi *et al.*, "Data flow orb-slam for real-time performance on embedded gpu boards," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 5370–5375.
[41] X. Chen, A. Milioto, E. Palazzolo *et al.*, "Suma++: Efficient lidar-based semantic SLAM," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 4530–4537.
[42] R. Li, S. Wang, Z. Long *et al.*, "Undeepvo: Monocular visual odometry through unsupervised deep learning," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 7286–7291.

Lia Morra received the M.Sc. and the Ph.D. degrees in computer engineering from Politecnico di Torino, Italy, in 2002 and 2006. Currently, she is assistant professor the Dip. di Automatica e Informatica of Politecnico di Torino. She is serving as Associated Editor for the IEEE Consumer Electronics Magazine. Her research interests include computer vision, pattern recognition, and machine learning.



Andrea Biondo received the M.Sc. in Mechatronics engineering and 2^{nd} level specializing master in AI&Cloud from Politecnico di Torino, Italy, in 2020 and 2021. He is currently working in Logistics Reply as a Machine Learning Engineer on time-series forecasting and computer vision applications.

Nicola Poerio received the M.Sc. degree in mechanical engineering from Politecnico di Milano, Italy, in 2005. Currently, he is Head of Prediction and Motion Planning Algorithms in the AI and Autonomous Driving dept. at Stellantis. His topics of interest include deep learning and robotics.

Fabrizio Lamberti received his M.Ss. and his Ph.D. degrees in computer engineering from Politecnico di Torino, Italy, in 2000 and 2005. He is now a full professor at Politecnico di Torino. His research interests include computer graphics, human-machine interaction and intelligent systems. He is serving as Associate Editor for IEEE Transactions on Consumer Electronics, IEEE Transactions on Computers, IEEE Transactions on Learning Technologies, and IEEE Consumer Electronics Magazine.