Journal of Field Robotics

WILEY

RESEARCH ARTICLE OPEN ACCESS

From "Mirror Flower, Water Moon" to Multi-Task Visual Prospective Representation Learning for Unmanned Aerial Vehicles Indoor Mapless Navigation

Yingxiu Chang[1] | Yongqiang Cheng[2] | John Murray[2] | Muhammad Khalid[1] | Umar Manzoor[3]

[1]School of Computer Science, University of Hull, Kingston upon Hull, England | [2]Faculty of Business & Technology, University of Sunderland, Sunderland, England | [3]Faculty of Science & Engineering, University of Wolverhampton, Wolverhampton, England

Correspondence: Yongqiang Cheng (yongqiang.cheng@sunderland.ac.uk)

Received: 5 June 2024 | **Revised:** 20 June 2025 | **Accepted:** 14 August 2025

Funding: The work has been supported in part by China Scholarship Council (202008010003).

Keywords: contrastive learning | indoor unknown environment | mapless navigation | prospective classification-aware representation | prospective regression-aware representation | UAV | visual prospective representation learning (VPRL)

ABSTRACT

Vision-based deep learning models have been widely adopted in autonomous agents, such as unmanned aerial vehicles (UAVs), particularly in reactive control policies that serve as a key component of navigation systems. These policies enable agents to respond instantaneously to dynamic environments without relying on pre-existing maps. However, there remain open challenges to improve the agent's reactive control performance: (1) Is it possible and how to anticipate future states at the current moment to benefit control precision? (2) Is it possible and how can we anticipate future states for different sub-tasks when the agent's control consists of both discrete classification and continuous regression commands? Inspired by the Chinese idiom "Mirror Flower, Water Moon," this paper hypothesizes that future states in the latent space can be learnt from sequential images using contrastive learning, and consequently proposes a light-weight Multi-task Visual Prospective Representation Learning (MulVPRL) framework for benefiting reactive control. Specifically, (1) This paper leverages the advantage of contrastive learning to correlate the representations obtained from the latest sequential images and one image in the future. (2) This paper constructs an integrated loss function of contrastive learning for classification and regression sub-tasks. The MulVPRL framework outperforms the benchmark models on the public HDIN and DroNet datasets, and obtained the best performance in real-world experiments (46.9 m, 177svs. SOTA 27.3 m, 136 s). Therefore, the multi-task contrastive learning of the light-weight MulVPRL framework enhances reactive control performance on a 2D plane, and demonstrates the potential to be integrated with various intelligent strategies, and implemented on ground vehicles.

1 | Introduction

"Mirror Flower, Water Moon" is a Chinese idiom (Wang 2006) that is metaphorically used to depict something that can be seen but is untouchable. Just like the moon reflected in the water shown in Figure 1a which is the illusion of the real world. Inspired by this idiom, future state anticipation is likewise the illusion of the present moment like the old man reflected in the mirror in Figure 1b. The water and mirror surfaces can be regarded as a latent space containing visual representations that are encoded from reality and decoded towards illusions. Unlike recent visual-based control strategies that immediately respond to the latest observations relying on learning present representations (Yang et al. 2019, 2021; Chang et al. 2023b), this paper intends to study the possibility of anticipating future states based on present visual data and consequently advancing the reactive control performance for autonomous agents.

(a) "Mirror Flower, Water Moon"

(b) Magic mirror

FIGURE 1 | (a) Reflected moon in the water is untouchable. (b) A magic mirror that can foresee the future (this is produced by ChatGPT 4o OpenAI 2024). (a) "Mirror Flower, Water Moon", (b) Magic mirror. [Color figure can be viewed at wileyonlinelibrary.com]

Anticipating future states is to learn a representation from the latest data inputs during time $t-n$ to t to denote the information extracted from the $t+n$ future data. Josselyn and Tonegawa (Josselyn and Tonegawa 2020) indicated that rodent brains existing "Memory Engram" have the potential to recall the past and anticipate the future through Artificial Intelligence (AI)-based technologies. According to their persuasive evidence, Visual Prospective Representation Learning (VPRL) used for anticipating future visual states has been initially studied for video action prediction (Vondrick et al. 2016; Jain et al. 2016; Surís et al. 2021), future frame generation (Zeng et al. 2017), pedestrian path generation (Park et al. 2016) and robotic arm reactive control (Koppula and Saxena 2015), which all have demonstrated better performance compared to other methods without VPRL.

However, VPRL has rarely been implemented in autonomous systems, particularly those systems for UAV indoor navigation. Although deep reinforcement learning (Doukhi and Lee 2022; Xie et al. 2020), depth image prediction (Kouris and Bouganis 2018; Chakravarty et al. 2017; Yang et al. 2019, 2021) and relative position recognition (Padhy et al. 2018; Chhikara et al. 2021) have achieved navigation in unknown spaces without collision, they can solely be treated as short-term reactive control policies since regardless of long-term historical information. Recently, short-term policies serving as agent-decision module of unmanned ground vehicles (UGVs) autonomous systems are not just integrated with map prediction (Chaplot et al. 2020; Ramakrishnan et al. 2020), but also can be collaborated with image retrieval and localization (Wei et al. 2024; Balntas et al. 2018; Leyva-Vallina et al. 2023, 2024; Laskar et al. 2017) to record long-term historical information, thus showing potential to achieve indoor mapless navigation without 2D/3D map-building (Güzel 2013; Chang et al. 2023a). Since the short-term reactive policy becomes a key component of indoor mapless navigation systems and requires a light-weight architecture to obtain faster processing speed and higher control precision, VPRL has the potential to satisfy and is the focus of this paper.

In recent years, contrastive learning has been applied for many real-world applications such as classification tasks of agriculture monitoring (Güldenring and Nalpantidis 2021), healthcare management (Tang et al. 2021), emotion recognition (Song et al. 2022) and video action classification (Pan et al. 2021; Jain et al. 2016), and regression tasks of UAV reactive control for drone racing (Fu et al. 2023). Different to the direct label-representation association of supervised learning results in extracting the most relevant features, contrastive learning distinguishes similar and dissimilar samples by comparing representations based on defining appropriate similarity scores. Hence, contrastive learning can learn more weak relevant features (Koohpayegani et al. 2020; Robinson et al. 2021) which can benefit models' prediction performance (Yu and Liu 2004; Chang et al. 2023b).

Due to the advantages of contrastive learning and the potential model improvement based on VPRL, this paper proposes a novel Multi-task Visual Prospective Representation Learning (MulVPRL) framework to simultaneously obtain the prospective classification-aware and regression-aware representations from the latest sequential images. The MulVPRL framework presents competitive performance on two public datasets. The real-world experiments depend on a Nano-size UAV (Bitcraze 2019) as the practical agent shown in Figure 2, which achieves the best control performance. This framework with light-weight architecture serves as a short-term control policy for instantaneous reaction and subsequently can be integrated with other intelligent policies to record long-term historical information for accomplishing complicated tasks such as victim search-and-rescue in unknown spaces based on suitable agent platforms. The main contributions of this paper are summarized below, and the trained model, code and video are publicly released on GitHub (Chang 2024).

- This paper proposes a novel MulVPRL framework by developing appropriate similarity measurements for the associated contrastive loss functions to respectively contrast the present regression-related and classification-related representations with the corresponding actual future representations in the latent space.
- The visual prospective representations learned by the MulVPRL framework are specifically mapped to the UAV reactive control in this paper, and have been verified and compared with relevant models on both public datasets and in real-world environments.

The rest of this paper is organized as follows: Section 2 introduces the related works of representation learning and indoor navigation strategies based on different agent platforms. Section 3 illustrates the relevant theoretical principles of the methods described in Section 4. Sections 5 and 6, respectively, demonstrate the evaluation on public datasets and real-world experiments, which both include visualizations and discussions. Finally, this paper will be concluded in Section 7.

2 | Related Works

This section will firstly introduce representation learning background regarding contrastive classification and regression along with prospective representation learning. Then, it will provide investigations of indoor navigation strategies.

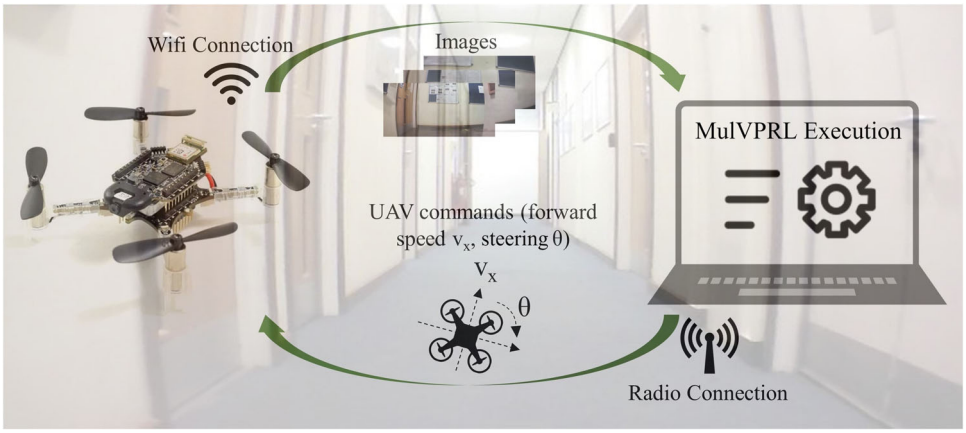


FIGURE 2 | Real-world experiments. The Nano-UAV sends the sequential greyscaled images to the laptop via WiFi connection while the laptop executes the MulVPRL framework and sends back the UAV's control commands including forward v_x and steering angular velocities θ via radio connection. The UAV automatically navigates in practical environments. [Color figure can be viewed at wileyonlinelibrary.com]

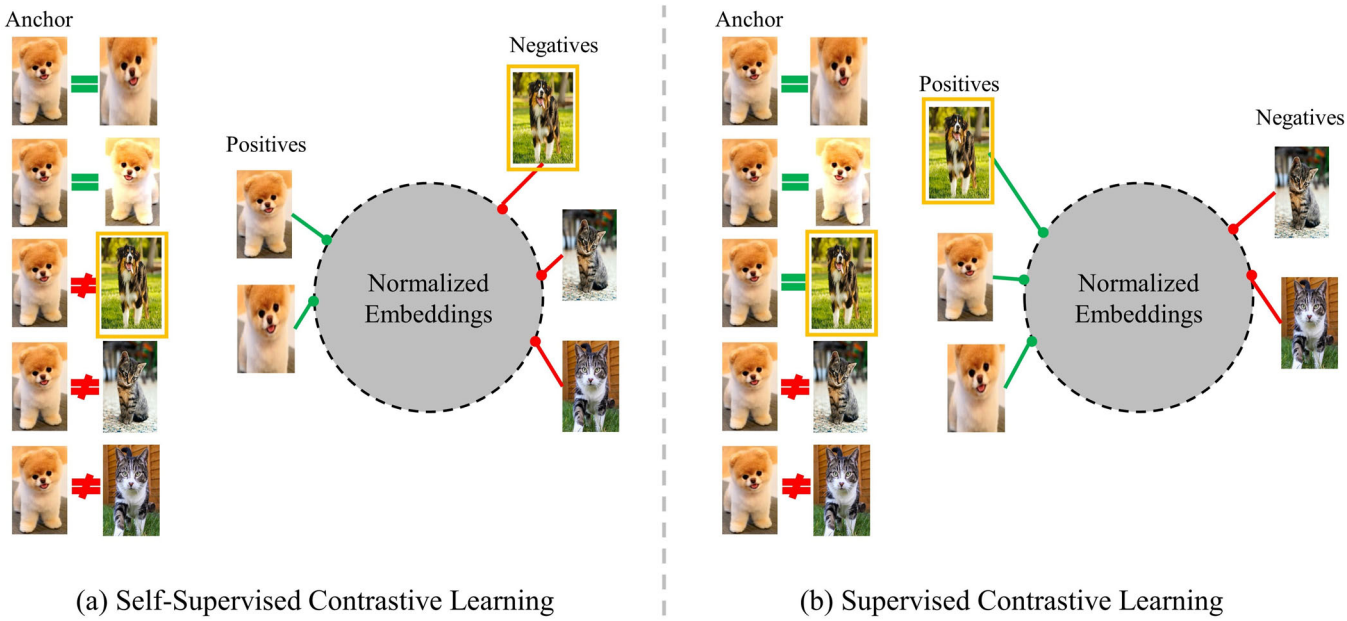


FIGURE 3 | (a) Self-supervised contrastive learning. The dog in the yellow box is defined as negative sample since it is not generated from the same anchor image, hence it might be erroneously pushed towards the cat's group. (b) Supervised contrastive learning. The dog image in the yellow box is defined as a positive sample since it belongs to the same class of the anchor image and hence, has been correctly clustered into the dog's group. [Color figure can be viewed at wileyonlinelibrary.com]

2.1 | Contrastive Classification

The fundamental of contrastive learning is to push away the negative samples and pull together the positive samples. If the answer to "Is it from the same image" is yes, the newly augmented images are positive samples of the original image, otherwise they're negative samples. This is called self-supervised contrastive learning without using actual labels (He et al. 2020; Chen et al. 2020) to separate representations that are obtained from the encoder and hence, benefit the following MLP layers for image classification such as releasing tiresome labeling works on videos (Pan et al. 2021). Unfortunately, self-supervised contrastive learning sometimes presents mis-classifications as shown in Figure 3a relating to the insufficient definition regarding positive and negative samples. Therefore, Khosla et al. (Khosla et al. 2020) changed the idea of "Is it from the same image" to "Is it from the same class" by introducing the actual class labels into the contrastive loss function. It is named Supervised Contrastive Learning (SupCon) as shown in Figure 3b, which obtained better classification performance than the self-supervised contrastive learning methods.

2.2 | Contrastive Regression

Due to the improved performance of contrastive classification, researchers have started to explore contrastive learning for regression tasks in which the continuous samples and labels have no discrete classes, such as activity scoring (Yu et al. 2021),

data set annotation (Ruan and Wang 2021) and face gaze estimation (Wang et al. 2022). In addition, several papers focusing on outdoor autonomous driving demonstrated that their vehicle steering prediction as agent decision based on SupCon (Khosla et al. 2020) is close to the performance based on supervised learning. For instance, Zhang et al. (2022) utilized a constant threshold to distinguish positive and negative samples according to the L1 distance between labels. Zheng et al. (2022) formed the consecutive labels as a vector to calculate the cosine similarity for measuring the distance between sample pairs and thus, can be used to identify positive and negative samples. Although these contrastive learning methods (Zhang et al. 2022; Zheng et al. 2022) initially addressed the challenge of defining positive and negative pairs for regression samples and obtained similar results of supervised learning, they ignored the relevance between consecutive samples. The relevance between consecutive samples, that is the intrinsic order relationship, is shown in Figure 4. Therefore, Zha et al. (Zha et al. 2024) proposed a Rank-N-Contrast model based on the "Ranking" idea for Supervised Contrastive Regression (SupCR) and outperformed other supervised and contrastive learning methods. Furthermore, the SupCR method was also verified to predict the ratio of steering angular velocity for outdoor ground vehicles and indoor UAVs reactive control by Chang et al. (2023b). The significant performance improvement of single-image contrastive regression achieved by Zha et al. (2024) and Chang et al. (2023b) serves as the motivation to further enhance the model's performance by incorporating spatial-temporal information and VPRL.

2.3 | Prospective Representation Learning

Depending on the mutual representations that are respectively encoded from images at time t and in the future, Vondrick and Zeng et al. (Vondrick et al. 2016; Zeng et al. 2017) generated the visual prospective representation for video action classification and future frame generation. Moreover, the Convolutional Neural Networks with Recurrent Neural Networks (CNN +RNN) structure extracts the spatial-temporal representation by taking the image sequence from the past. Oord et al. (van den Oord et al. 2019) indicated that the spatial-temporal representation contains sufficient information to predict the future. Learning prospective representations based on the CNN+RNN structure has been proved for classification tasks such as audio recognition (Haresamudram et al. 2021) and video action prediction (Surís et al. 2021; Jain et al. 2016), and recently applied for reactive control of UGV autonomous navigation (He et al. 2023). However, none of them have coped with regression tasks, and even multi-task applications which include classification and regression sub-tasks to smooth the agent, especially the UAV reactive control for indoor mapless autonomous navigation.

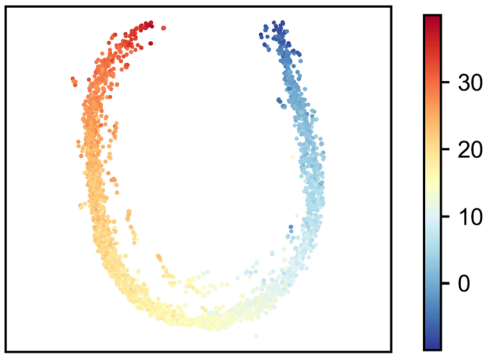


FIGURE 4 | Intrinsic order relationship between consecutive samples. It is plotted by Umap visualization technology (McInnes et al. 2018). The *X*- and *Y*-axes represent the coordinates of the data in the 2D XY space after dimensionality reduction. The axes do not have direct physical concepts but rather are used to depict the relationship between data. The representation distribution according to the associated actual value of labels from low → high corresponds to colors from cold → warm. (Zha et al. 2024). [Color figure can be viewed at wileyonlinelibrary.com]

Therefore, this paper innovatively leverages the advantage of contrastive learning into multi-task VPRL and applies the commonly used CNN+RNN structure from contrastive and VPRL methods (Surís et al. 2021; Jain et al. 2016; van den Oord et al. 2019; Haresamudram et al. 2021; He et al. 2023; Pan et al. 2021) as the backbone of the MulVPRL framework.

2.4 | Indoor Autonomous Navigation

The workflow of the agent's conventional navigation system covers environmental perception, map-building, self-localization and path planning (Khairuddin et al. 2015). Simultaneous Localization and Mapping (SLAM) such as visual SLAM (Mur-Artal et al. 2015; Mur-Artal and Tardós 2017; Engel et al. 2014) and Light Detection and Ranging (LiDAR) SLAM (Hornung et al. 2013; Batinovic et al. 2021) is the prime subcomponent for goal-directional navigation in large-scale indoor environments (Chen et al. 2022; Youn et al. 2021; Esrafilian and Taghirad 2016). Although the SLAM-based algorithms can plan the optimal path to the destination based on the continuously updated map, the overall workflow is computational-demanding resulting in massive time-consumption (Chang et al. 2023a).

To improve real-time control performance, researchers explored mapless navigation by applying CNN models to perceive environmental information. For instance, processing event images (Vemprala et al. 2021), optical flow depth estimation (McGuire et al. 2017) and RGB image depth estimation (Chakravarty et al. 2017; Yang et al. 2019, 2021) allow the following controller to generate UAV commands accordingly. Moreover, CNN models are able to directly map the captured images to UAV commands based on supervised (Loquercio et al. 2018; Palossi et al. 2019) and reinforcement learning (Singla et al. 2019; Fu et al. 2023; Xue and Gonsalves 2023). Although these strategies offer instantaneous reactions to control the UAV, they lack destination and long-term historical information and hence can only be treated as short-term control policies. Recently, there have been two solutions to address this limitation for long-term navigation. Firstly, capturing more surrounding environmental information by additionally using a backward camera or a 360° camera along with long-term image sequences, such as the UGV mapless navigation systems (Morin et al. 2023;

He et al. 2023). Another solution controls the UGV relying on a light-weight short-term policy as the agent-decision module, and periodically invoking a long-term policy with map-prediction at a constant interval (Chaplot et al. 2020; Ramakrishnan et al. 2020). In comparison, the short-term policy of the latter solution is important to ensure real-time responsiveness and requires higher control precision. Otherwise, the agent can be controlled into unexpected spaces, leading to more iterations and time-consumption for invoking long-term policies, which have negative impacts on specific tasks such as victim search-and-rescue.

3 | Theoretical Principles

This section respectively introduces the theoretical principles of VPRL and supervised contrastive learning, providing a theoretical foundation for the methodologies in subsequent MulVPRL framework.

3.1 | Predictive Coding of VPRL

The following Equations (1)–(3) indicate the theoretical principle summarized from predictive coding (van den Oord et al. 2019). In Equation (1), x_{t-n} to x_t represents the latest sequential images. *STEnc* is a spatial-temporal encoder composed of shared-weights CNN encoders followed by RNN layers (i.e. CNN+RNN structure). x_{t+n} in Equation (2) denotes a future image while *Enc* is a single CNN encoder that is same as those in *STEnc*. The objective of predictive coding of VPRL is to ensure that the representations z'_t and z_{t+n}, respectively, extracted from the latest sequential and future images have the greatest similarity, that is the cosine similarity approaching 1 shown as Equation (3).

$$z'_t = ST\ Enc(x_{t-n}, ..., x_t), \tag{1}$$

$$z_{t+n} = Enc(x_{t+n}), \tag{2}$$

$$sim\left(z'_t, z_{t+n}\right) = \frac{z'_t \cdot z_{t+n}}{||z'_t||||z_{t+n}||} \rightarrow 1. \tag{3}$$

3.2 | Supervised Contrastive Classification and Regression

According to the reference (Khosla et al. 2020), the summarized Equations (4)–(6) are used to demonstrate the theoretical principle of supervised contrastive classification. The index i is called the anchor and B is a set of all sample indices in one batch.

In Equation (4), P_i is the set including indices of all positive samples of the anchor sample in a batch. When sample j shares the same class label as the anchor sample ($y_i = y_j$), its index j will be stored into P_i.

In Equation (5), $(z_i \cdot z_j)$ and $(z_i \cdot z_k)$ calculate the cosine similarity between two representations. τ is the temperature parameter for controlling the penalty on hard negative samples which have greater similarity. The anchor sample pairs up with its all positive samples for calculating the summation shown as $\sum_{j\in P_i} exp((z_i \cdot z_j)/\tau)$. Then, the anchor sample pairs up with all samples in the same batch to calculate the summation shown as $\sum_{k\in B} exp((z_i \cdot z_k)/\tau)$ for contributing to the denominator. Subsequently, $\mathcal{L}_{SupCon_i}$ calculates the average loss of the anchor sample by dividing $|P_i|$, i.e. the number of positive samples of the anchor sample.

Finally, since each sample in the same batch will be the anchor once for calculating the summation shown as $\sum_{i\in B}\mathcal{L}_{SupCon_i}$, the total SupCon loss $\mathcal{L}_{SupCon_{total}}$ is obtained by dividing the batch size $|B|$ in Equation (6). Minimizing the $\mathcal{L}_{SupCon_{total}}$ via introducing class labels to the loss function allows to pull together the representations from the same class and push away the representations from different classes.

$$P_i = \left\{j | y_i = y_j\right\}, \tag{4}$$

$$\mathcal{L}_{SupCon_i} = \frac{1}{|P_i|}\sum_{j\in P_i} -\log\frac{\exp((z_i \cdot z_j)/\tau)}{\sum_{k\in B}\exp((z_i \cdot z_k)/\tau)}, \tag{5}$$

$$\mathcal{L}_{SupCon_{total}} = \frac{1}{|B|}\sum_{i\in B}\mathcal{L}_{SupCon_i}. \tag{6}$$

Similarly, the summarized Equations (7–9) are used to demonstrate the theoretical principle of supervised contrastive regression inspired by the "Ranking" idea of reference (Zha et al. 2024).

Firstly, when anchor sample i and sample j form a sample pair, the index k will be stored into the N_{ij} set if the L1 distance of labels between samples i and k is greater or equal to that between samples i and j, that is $d(y_i, y_k) \geq d(y_i, y_j)$. This process is shown in Equation (7) and indicates that each sample pair has a different number of relevant negative pairs.

In Equation (8), for each sample pair formed with anchor sample i, the summation of relevant negative pairs including itself contributes to the denominator shown as $\sum_{k\in N_{ij}} exp((z_i \cdot z_k)/\tau)$. Since each sample in the same batch will be the anchor once and paired up with all samples, the total SupCR loss $\mathcal{L}_{SupCR_{total}}$ is calculated as Equation (9) by dividing $|B|^2$, that is the square of the batch size. Minimizing the $\mathcal{L}_{SupCR_{total}}$ will sort all representations according to the magnitude of regression labels, that is capturing the intrinsic ordered relationship between consecutive samples.

$$N_{ij} = \left\{k | d(y_i, y_k) \geq d(y_i, y_j)\right\}, \tag{7}$$

$$\mathcal{L}_{SupCR_{ij}} = -\log\frac{\exp((z_i \cdot z_j)/\tau)}{\sum_{k\in N_{ij}}\exp((z_i \cdot z_k)/\tau)}, \tag{8}$$

$$\mathcal{L}_{SupCR_{total}} = \frac{1}{|B|^2}\sum_{i\in B}\sum_{j\in B}\mathcal{L}_{SupCon_{ij}}. \tag{9}$$

In general, the theoretical principles described in this section provide a fundamental understanding of the methods in the following Section 4.

4 | Methods

This section will present the proposed novel MulVPRL framework as well as detailing two training stages along with the proposed loss functions. The agent in the following contents will be embodied as a UAV.

4.1 | MulVPRL Framework—Training Stage I

The 1st training stage is to update the weights within the CNN + RNN structure for learning the visual prospective regression-aware and classification-aware representations. The theoretical objective is to simultaneously correlate the prospective regression-aware and classification-aware representations obtained from the latest observed sequential images between x_{t-kn} and x_t, with the corresponding spatial representation of the future image x_{t+1n}. The t, k and n respectively represent the current time, the number of previously observed images and the image sampling step.

As shown in Figure 5, the CNN + RNN structure consists of parallel shared-weights encoders (*Enc*), an *RNN* layer, and Regression and Classification Projectors (*RegProj* and *ClsProj*). Specifically, the final flattened layer of the ResNet8 (Loquercio et al. 2018; Palossi et al. 2019) shared-weights encoder outputs the spatial representation z with the dimension of 128. Then, the *RNN* layer outputs spatial-temporal representation c_t with the dimension of 256 by feeding sequential spatial representations between $z_{t-kn} \cdots z_t$. Subsequently, the regression and classification projectors generate the prospective regression-aware z'_{reg} and classification-aware representations z'_{cls} with the dimension of 128, which are the same dimension as the spatial representations between $z_{t-kn} \cdots z_{t+1n}$. Finally, the MulVPRL loss function contains SupCR and SupCon loss functions to calculate the total loss value $\mathcal{L}_{MulVPRL}$ for updating the weights of shared-weights encoders, RNN layer and projectors shown in the dashed box in Figure 5.

Algorithm 1 Training Stage I of the MulVPRL Framework

Definition: A batch contains B image sequences $[X]^B = [x_{t-kn}, \ldots x_{t+1n}]^B$ in which there are R regression-related sequences and C classification-related sequences. $R + C = B$.

Input: $[X]^B$, an example batch with image sequences

1: $[Z]^B = Enc([X]^B)$, in which $[Z]^B = [z_{t-kn}, \ldots, z_{t+1n}]^B$;

2: $[c_t]^B = RNN([Z_t]^B)$, $[Z_t]^B = [z_{t-kn}, \ldots, z_t]^B$;

3: $[z'_{reg}]^B = RegProj([c_t]^B)$, $[z'_{cls}]^B = ClsProj([c_t]^B)$;

4: $\mathcal{L}_{SupCR} = f_{SupCRLoss}([z'_{reg}]^R, [z_{t+1n_{reg}}]^R, [y_{s_t}]^R)$,
$\mathcal{L}_{SupCon} = f_{SupConLoss}([z'_{cls}]^C, [z_{t+1n_{cls}}]^C, [y_{c_t}]^C)$;

5: $\mathcal{L}_{MulVPRL} = f_{MulVPRLLoss}(\mathcal{L}_{SupCR} + \mathcal{L}_{SupCon})$;

6: Updated weights of CNN+RNN structure by using $\mathcal{L}_{MulVPRL}$.

The procedures of the 1st training stage are presented in Algorithm 1 for each batch containing B image sequences. Upon processing by the paralleled shared-weights encoders (*Enc*), an *RNN* layer, and regression and classification projectors (*RegProj* and *ClsProj*) within the first three steps, this batch yields B prospective regression-aware $[z'_{reg}]^B$ and classification-aware representations $[z'_{cls}]^B$. Since each batch is composed of R regression-related and C classification-related sequences ($R + C = B$), $[z'_{reg}]^R$ and $[z'_{cls}]^C$ representations respectively contrast with corresponding future representations $[z_{t+1n_{reg}}]^R$ and $[z_{t+1n_{cls}}]^C$ in step 4. The SupCR $f_{SupCRLoss}$, SupCon $f_{SupConLoss}$ and MulVPRL $f_{MulVPRLLoss}$ loss functions calculated in steps 4 and 5 will be illustrated as follows.

The relative positive and negative samples for the anchor sample define the relationship between the anchor and

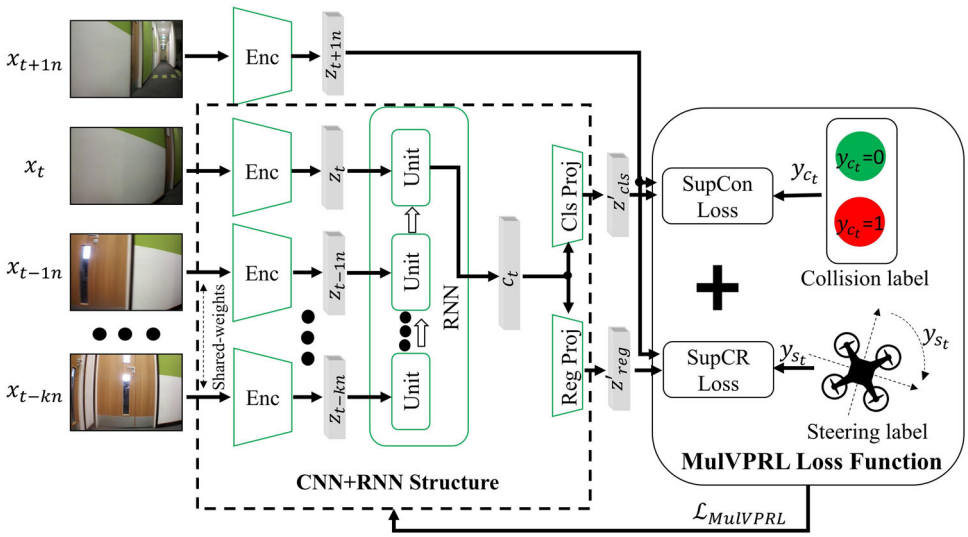


FIGURE 5 | The workflow of the 1st training stage of MulVPRL framework. The green modules denote that their weights are updated during this process. The backbones of the shared-weights encoder and RNN are, respectively, a ResNet8 model and a single gated recurrent unit (GRU) layer. Both Reg Proj and Cls Proj consist of a single-layer Fully Connected Neural Network (FCNN) while the total MulVPRL loss function includes SupCR and SupCon loss functions. [Color figure can be viewed at wileyonlinelibrary.com]

other samples which is essential for contrastive learning. According to the theoretical principles in Section 3, Figure 6 is used to better illustrate the SupCR and SupCon loss functions corresponding to the VPRL for steering-related and collision-related sequences.

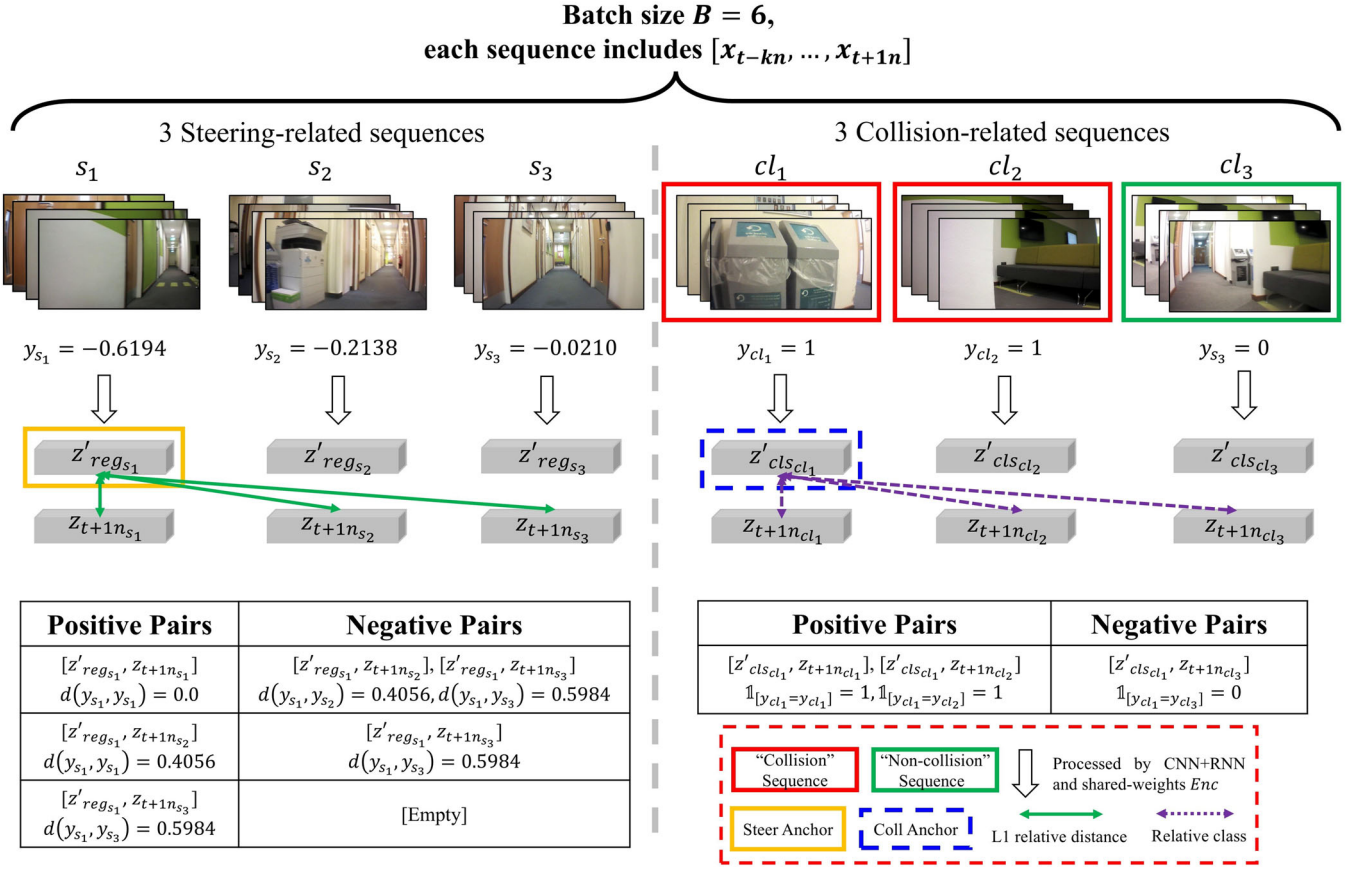
VPRL for Steering-Related Sequences

As shown in Figure 6, the $z'_{reg_{s_1}}$ representation marked by a yellow solid box is initially assumed as the "Steer Anchor". Then, the "Steer Anchor" pairs up with all steering-related future representations to form the "Positive Pairs" shown in the left table. According to the aforementioned theoretical principles of SupCR in Section 3.2, the negative pairs relevant to each positive pair have greater L1 relative distance of labels. For instance, the relevant negative pairs corresponding to the $[z'_{reg_{s_1}}, z_{t+1n_{s_1}}]$ positive pair are $[z'_{reg_{s_1}}, z_{t+1n_{s_2}}]$ and $[z'_{reg_{s_1}}, z_{t+1n_{s_3}}]$ since $d(y_{s_1}, y_{s_2})$ and $d(y_{s_1}, y_{s_3}) > d(y_{s_1}, y_{s_1})$. As a result, the contrastive loss to the "Steer Anchor" $z'_{reg_{s_1}}$ is $\frac{1}{R}\sum_{s_j,j=1}^{s_{n,n=R}} l_{s_i,s_j}$ where $i = 1$. The l_{s_i,s_j} is calculated according to Equation (10). Moreover, since each steering-related prospective representation in the same batch will be the anchor once, the total SupCR loss $\mathcal{L}_{SupCR}$ is computed as Equation (11).

$$l_{s_i,s_j} = -log\frac{exp\left(\left(z'_{reg_{s_i}} \cdot z_{t+1n_{s_j}}\right)/\tau\right)}{\sum_{s_k,k=1}^{s_{n,n=R}} \mathbb{1}_{\left[d(y_{s_i},y_{s_k}) \geq d(y_{s_i},y_{s_j})\right]} exp\left(\left(z'_{reg_{s_i}} \cdot z_{t+1n_{s_k}}\right)/\tau\right)}, \tag{10}$$

$$\mathcal{L}_{SupCR} = \frac{1}{B}\sum_{s_i,i=1}^{s_{n,n=R}} \frac{1}{R}\sum_{s_j,j=1}^{s_{n,n=R}} l_{s_i,s_j}. \tag{11}$$

s_i is the index of steering-related prospective representations, s_j, s_k and s_n represent the indices of steering-related future representations. R is the number of steering-related sequences in a batch with the size of B. $(z'_{reg_{s_i}} \cdot z_{t+1n_{s_j}})$ and $(z'_{reg_{s_i}} \cdot z_{t+1n_{s_k}})$ are responsible for computing the cosine similarity between the unit vectors which have the same dimension. $\mathbb{1}_{[d(y_{s_i},y_{s_k}) \geq d(y_{s_i},y_{s_j})]} = 1$ if the condition in the square bracket is true, otherwise equals 0. In summary, minimizing $\mathcal{L}_{SupCR}$ not only enhances the similarity between the regression-aware prospective representation with the actual future representation, but also sorts these prospective representations according to label numerical magnitudes for capturing their intrinsic ordered relationships.



Positive Pairs	Negative Pairs
$[z'_{reg_{s_1}}, z_{t+1n_{s_1}}]$ $d(y_{s_1}, y_{s_1}) = 0.0$	$[z'_{reg_{s_1}}, z_{t+1n_{s_2}}]$, $[z'_{reg_{s_1}}, z_{t+1n_{s_3}}]$ $d(y_{s_1}, y_{s_2}) = 0.4056, d(y_{s_1}, y_{s_3}) = 0.5984$
$[z'_{reg_{s_1}}, z_{t+1n_{s_2}}]$ $d(y_{s_1}, y_{s_1}) = 0.4056$	$[z'_{reg_{s_1}}, z_{t+1n_{s_3}}]$ $d(y_{s_1}, y_{s_3}) = 0.5984$
$[z'_{reg_{s_1}}, z_{t+1n_{s_3}}]$ $d(y_{s_1}, y_{s_3}) = 0.5984$	[Empty]

Positive Pairs	Negative Pairs
$[z'_{cls_{cl_1}}, z_{t+1n_{cl_1}}]$, $[z'_{cls_{cl_1}}, z_{t+1n_{cl_2}}]$ $\mathbb{1}_{[y_{cl_1}=y_{cl_1}]} = 1, \mathbb{1}_{[y_{cl_1}=y_{cl_2}]} = 1$	$[z'_{cls_{cl_1}}, z_{t+1n_{cl_3}}]$ $\mathbb{1}_{[y_{cl_1}=y_{cl_3}]} = 0$

FIGURE 6 | An example batch includes 3 steering-related (indices s_1 to s_3) and 3 collision-related (indices cl_1 to cl_3) sequences with their steering regression (y_{s_1} to y_{s_3}) and collision classification (y_{cl_1} to y_{cl_3}) labels at time t. Each sequence in the example batch obtains corresponding prospective representations (z'_{reg} and z'_{cls}) via the CNN+RNN structure, and future representations (z_{t+1n}) via the shared-weights encoder. The "Steer Anchor" $z'_{reg_{s_1}}$ and "Coll Anchor" $z'_{cls_{cl_1}}$ form positive and negative pairs with future representations, respectively, based on the L1 relative distance of steering labels (green solid double arrows) and relative class of collision labels (purple dashed double arrows). [Color figure can be viewed at wileyonlinelibrary.com]

VPRL for Collision-Related Sequences

As for collision-related sequences, the $z'_{cls_{cl_1}}$ representation is initially denoted as the "Coll Anchor" highlighted in a blue dashed box in Figure 6. According to the theoretical principles of SupCon in Section 3.2, the positive samples to the "Coll Anchor" are selected from the collision-related future representations between $z_{t+1n_{cl_1}}$ to $z_{t+1n_{cl_3}}$ if their sequences have the same collision label as the cl_1 sequence. Therefore, $[z'_{cls_{cl_1}}, z_{t+1n_{cl_1}}]$ and $[z'_{cls_{cl_1}}, z_{t+1n_{cl_2}}]$ are positive pairs while $[z'_{cls_{cl_1}}, z_{t+1n_{cl_3}}]$ is negative pair shown in the right table since $\mathbb{1}_{[y_{cl_1}=y_{cl_1}]} = \mathbb{1}_{[y_{cl_1}=y_{cl_2}]} = 1$ but $\mathbb{1}_{[y_{cl_1}=y_{cl_3}]} = 0$. The contrastive loss to the "Coll Anchor" $z'_{cls_{cl_1}}$ is $\frac{1}{P_{cl_i}} \sum_{cl_j,j=1}^{cl_{n,n=C}} l_{cl_i,cl_j}$ where $i = 1$. The l_{cl_i,cl_j} is calculated based on Equation (12) and P_{cl_i} is obtained from Equation (13).

$$l_{cl_i,cl_j} = -log \frac{\mathbb{1}_{[y_{cl_i}=y_{cl_j}]} exp\left(\left(z'_{cls_{cl_i}} \cdot z_{t+1n_{cl_j}}\right)/\tau\right)}{\sum_{cl_k,k=1}^{cl_{n,n=C}} exp\left(\left(z'_{cls_{cl_i}} \cdot z_{t+1n_{cl_k}}\right)/\tau\right)}, \tag{12}$$

$$P_{cl_i} = \sum_{cl_j,j=1}^{cl_{n,n=C}} \mathbb{1}_{[y_{cl_i}=y_{cl_j}]}, P_{cl_i} \leq C, \tag{13}$$

$$\mathcal{L}_{SupCon} = \frac{1}{B} \sum_{cl_i,i=1}^{cl_{n,n=C}} \frac{1}{P_{cl_i}} \cdot \sum_{cl_j,j=1}^{cl_{n,n=C}} l_{cl_i,cl_j}. \tag{14}$$

cl_j, cl_k and cl_n are the indices of the collision-related future representations and cl_i specifically indicates the index of collision-related prospective representation. C shows the number of collision-related sequences in the same batch. P_{cl_i} determines the number of positive samples corresponding to the anchor $z'_{cls_{cl_i}}$. Since each collision-related prospective representation in the same batch will be the anchor once, Equation (14) firstly computes the average loss of positive pairs that are formed with the anchor $z'_{cls_{cl_i}}$ and subsequently calculates the total SupCon loss $\mathcal{L}_{SupCon}$.

Total MulVPRL Loss Function

Similar to the DroNet methods for training a multi-task model (Loquercio et al. 2018; Palossi et al. 2019), the following Equation (15) MulVPRL loss function which is composed of $\mathcal{L}_{SupCR}$ and $\mathcal{L}_{SupCon}$ loss values calculates the total MulVPRL loss value $\mathcal{L}_{MulVPRL}$ for conjunctly updating the weights of the CNN +RNN structure shown as the black dashed box in Figure 5. *epoch* represents the current training epoch while *decay* and $epoch_0$ will be specifically introduced in Section 5.1.

$$\mathcal{L}_{MulVPRL} = \mathcal{L}_{SupCR} + max(0, 1 - exp^{-decay(epoch-epoch_0)}) \mathcal{L}_{SupCon}. \tag{15}$$

4.2 | Mulvprl Framework—Training Stage II

As for the 2^{nd} training stage shown in Figure 7, the parameters within the CNN+RNN structure will be frozen. The separate Regression and Classification MLP layers (*RegMLP* and *ClsMLP*) composed of a single-layer FCNN can be trained for mapping the prospective regression-aware z'_{reg} and classification-aware representations z'_{cls} into steering value y'_{s_t} and collision probability y'_{c_t}. These two separate *MLP* layers are trained based on the Multi-task Supervised Learning (MulSL) loss function as Equation (16) shows, in which the total MulSL loss value $\mathcal{L}_{MulSL}$ combines Mean Square Error (MSE) $\mathcal{L}_{MSE}$ and Binary CrossEntropy (BCE) $\mathcal{L}_{BCE}$ losses. This MulSL loss function is the same as the multi-task DroNet model (Loquercio et al. 2018; Palossi et al. 2019).

$$\mathcal{L}_{MulSL} = \mathcal{L}_{MSE(y'_{s_t}, y_{s_t})} + max\Big(0, 1 - exp^{-decay(epoch-epoch_0)}\Big) \mathcal{L}_{BCE(y'_{c_t}, y_{c_t})}. \tag{16}$$

y'_{s_t} and y_{s_t} are the prediction and ground truth of steering values while y'_{c_t} and y_{c_t} respectively represent the prediction and

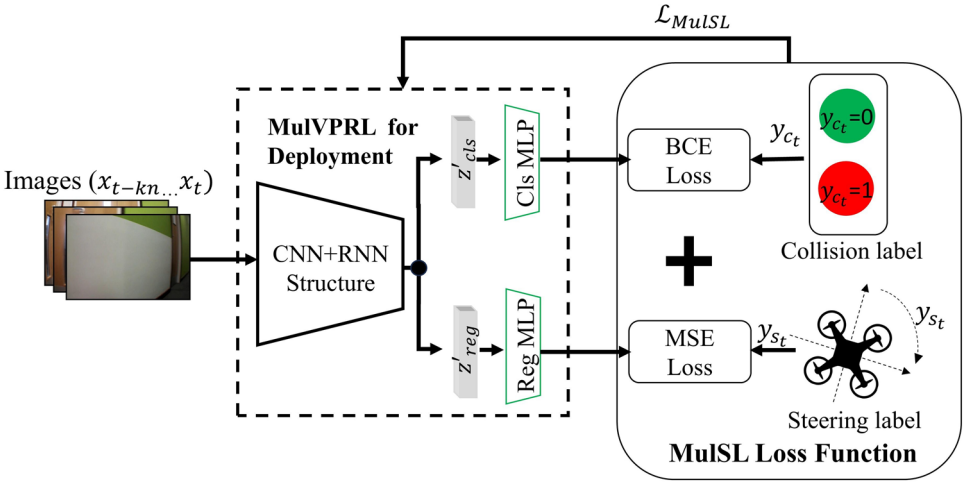


FIGURE 7 | The workflow of the 2nd training stage of the MulVPRL framework. The green modules, RegMLP and ClsMLP, are only updated during this process. All sub-components within the black dashed box are ultimately deployed for image inference. [Color figure can be viewed at wileyonlinelibrary.com]

ground truth for collision classification. *epoch* represents the current epoch during training while *decay* and $epoch_0$ will be specifically introduced in Section 5.1

5 | Evaluation on Datasets

This section will firstly introduce the public datasets and training setup of the MulVPRL framework. Then, the relevant models and the MulVPRL framework will be compared on the datasets. Ablation studies, visual demonstrations and statistical analyses are also included.

5.1 | Datasets and Training Setup

In recent years, numerous datasets have been developed to train models for controlling the UAV in indoor and outdoor environments. For instance, the model trained on the ICL data set (Kouris and Bouganis 2018) is able to estimate distances at directions $[-30°, 0°, 30°]$ within the front field of view. The ICL data set is more focused on reactive collision avoidance rather than navigational direction. Moreover, He et al. (2023) collected visual information by following the expert trajectory in large-scale virtual environments to form a data set and hence, enabling the model to achieve long-term navigation tasks. However, the collected 360° images are inappropriate for the commonly used UAVs with only a monocular camera.

The most appropriate datasets for evaluation are the DroNet (Loquercio et al. 2018; Palossi et al. 2019) and HDIN (Chang et al. 2022) UAV multi-task reactive control datasets, which both involve "Steering" and "Collision" subsets. The DroNet data set is entirely created in outdoor environments in which steering labels are ranged within $[-\frac{\pi}{2}, \frac{\pi}{2}]$ (Udacity 2016) while collision labels are 0 "Non-collision" and 1 "Collision" depending on the distance-to-collision. There are totally 64204 frames for training and 6855 frames for testing. The DroNet model trained on this data set was capable of controlling the UAV flying on outdoor vehicle roads and in indoor corridors and car parks.

The images and labels of the HDIN data set are completely collected on an actual UAV platform in indoor environments. The labels from the "Steering" subset are the Scalable Angular Velocity of label type 3 according to the original HDIN reference (Chang et al. 2022). These steering labels represent the ratio of maximum angular velocity within the range of $[-1, 1]$, which are also suitable for different UAV platforms. The labels from the "Collision" subset are manually annotated as 0 "Non-collision" and 1 "Collision" to represent collision probability. There are 10192 frames from 42 different trajectories for training with 1547 frames from 6 trajectories for testing in the "Steering" subset, and 3390 frames from 13 trajectories for training with 457 frames from 2 trajectories for testing in the "Collision" subset.

As for the training setup of the MulVPRL framework, Table 1 presents the hyperparameters for training on the HDIN and DroNet datasets based on the Adam optimizer (Kingma and Ba 2017).

5.2 | Quantitative Results on Datasets

The overview architectures of representative models with different learning methods are presented in Figure 8a–e for comparison with the MulVPRL framework shown in Figure 8f. The brief introductions of these models are described below.

DroNet+SL (Figure 8a): The baseline DroNet model with ResNet8 backbone was trained based on supervised learning for directly predicting the steering value and collision probability by processing the present image x_t.

TABLE 1 | Hyperparameters of the MulVPRL framework training on the HDIN and DroNet datasets.

Hyperparameters	On HDIN data set	on DroNet data set
Greyscaled image size	324 × 244	200 × 200
Initial learning rate of the first stage	$1e^{-3}$	$1e^{-3}$
Learning decay of the first stage	$1e^{-4}$	$1e^{-5}$
Epochs of the first stage	100	50
Initial learning rate of the second stage	$1e^{-3}$	$1e^{-3}$
Learning decay of the second stage	$1e^{-4}$	$1e^{-4}$
Epochs of the second stage	50	50
Temperature τ in Equations (10) and (12)	0.25	0.20
decay in Equations (15) and (16)	1/3	1/5
$epoch_0$ in Equations (15) and (16)	3	5
Batch size N	48	32
Sequence length	5	5
The number of historical images k in each sequence	3	3
Image sampling step n in each sequence	2	2

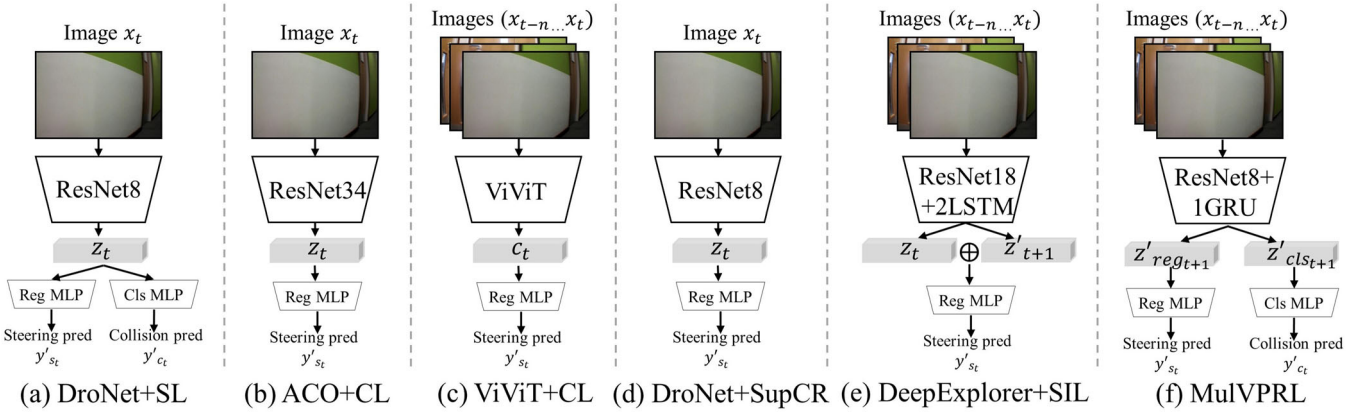


FIGURE 8 | Architectures of relevant models for comparison. z_t and c_t represent the spatial and spatial-temporal representations at time t. z'_{t+1}, $z'_{\text{reg}_{t+1}}$ and $z'_{\text{cls}_{t+1}}$ represent the prospective representations at time $t+1$. y'_{s_t} and y'_{c_t} are the predicted steering value and collision probability. $\oplus$ denotes representation concatenation. (a) DroNet+SL (b) ACO+CL (c) ViViT+CL (d) DroNet+SupCR (e) DeepExplorer+SIL (f) MulVPRL. [Color figure can be viewed at wileyonlinelibrary.com]

Nvidia CNN + SL: It serves as a baseline model (Bojarski et al. 2016). Its architecture is a 9-layer CNN training based on supervised learning to only predict steering values, which is similar to Figure 8b.

3D LSTM + SL: It serves as another baseline model (Du et al. 2019). It is composed of 3D convolutional layers followed by LSTM layers to extract the spatial-temporal representations based on supervised learning, and subsequently just predict the steering value.

ACO + CL (Figure 8b): Zhang et al. (2022) used a constant threshold to distinguish positive and negative samples of the anchor sample. It is called Action-Conditioned (ACO) Contrastive Learning and selected the ResNet34 as the encoder's backbone to only predict the steering value by processing the present image x_t.

ViViT + CL (Figure 8c): Zheng et al. (2022) transformed the consecutive actual steering labels into the unit vectors named "Positiveness" for dynamically weighting the contrastive loss. The encoder's backbone is the Video Vision Transformer (ViViT) (Arnab et al. 2021) which extracts the spatial-temporal representation from image sequence $x_{t-n}, ..., x_t$ for predicting the steering value.

DroNet + SupCR (Figure 8d): Similar to the DroNet model in Figure 8a, Chang et al. (Chang et al. 2023b) applied ResNet8 as the encoder's backbone but was trained based on SupCR (Zha et al. 2024) for only predicting the steering value from present image x_t.

DeepExplorer + SIL (Figure 8e): The DeepExplorer (He et al. 2023) applied ResNet18 as the encoder's backbone and followed with two Long Short-Term Memory (LSTM) layers. By concatenating the visual prospective representation z'_{t+1} generated from the last unit of the final LSTM layer with the actual representation z_t from the image at time t, the final MLP layer predicts the action $a \in [moveforward, turnleft, turnright]$ based on $a = MLP(z'_{t+1} \oplus z_t)$. For the reasons of fair comparison and the output of the original DeepExplorer is mainly used for deciding movement direction, the original loss function of Sparse Categorical Crossentropy for action classification is replaced by the MSE for steering regression. This replacement is denoted as Reg MLP in Figure 8e, $y'_{s_t} = RegMLP(z'_{t+1} \oplus z_t)$, and retrained the DeepExplorer based on the same supervised imitation learning.

Table 2 quantitatively shows the models' performance on the HDIN and DroNet datasets. All results are obtained from their original references or provided trained model (ACO+CL) except the DeepExplorer model which was validated on different datasets and requires retraining. A notable case is the steering prediction of the ViViT. Different from the common RMSE regression metric, ViViT adopted regression accuracy (results marked by " [7]") to assess performance, where a higher regression accuracy indicates better performance. Therefore, we also calculated the regression accuracy of the baseline DroNet model on the same data set for comparison. The best results of steering regression and collision classification on each data set are highlighted in bold while "-" means the compared models do not have this prediction.

In comparison between Figure 8a–c, ACO and ViViT models have more parameters and deeper architecture for extracting visual representation. However, their regression performance has not outperformed the baseline DroNet model which only uses ResNet8 as the backbone, and worse than the baseline Nvidia CNN and 3D LSTM models. The major reason lies in the limited definition of positive and negative pairs for contrastive learning. Specifically, the ACO+CL defines a constant threshold $\epsilon = 0.05$ to distinguish the positive and negative sample pairs based on the relative distance between labels, that is if the L1 distance of labels of two images is greater than ϵ, they are negative sample pairs. In addition, the ViViT + CL utilizes the unit vector of labels to calculate the similarity between sample pairs, that is the relevant negative pairs have smaller similarities. However, these two definitions fail to capture the intrinsic ordered relationship between consecutive samples and thus, result in suboptimal performance.

Moreover, according to the observation from Figure 8a,d, the DroNet+SupCR has the same encoder's backbone as the DroNet

TABLE 2 | Comparison of relevant models with different learning methods on the HDIN and DroNet datasets.

References	Models and learning methods	VPRL[4]	Results on HDIN data set		Results on DroNet data set	
			RMSE[5]	Acc.[6]	RMSE	Acc.
Loquercio et al. (2018) Palossi et al. (2019)	DroNet + SL[1]	×	0.123	85.8%	0.110/75.5%[7]	95.4%
Bojarski et al. (2016)	Nvidia CNN + SL	×	—	—	0.099	—
Du et al. (2019)	3D LSTM + SL	×	—	—	0.112	—
Zhang et al. (2022)	ACO + CL[2]	×	—	—	0.1571[8]	—
Zheng et al. (2022)	ViViT + CL	×	—	—	51.9%[7]	—
Chang et al. (2023b)	DroNet + SupCR	×	0.118	—	0.107	—
He et al. (2023)	* DeepExplorer + SIL[3]	✓	0.085	—	**0.091**	—
Ours	MulVPRL	✓	**0.083**	**86.6%**	0.098	**96.0%**

Note: SL[1]: supervised learning; CL[2]: contrastive learning; SIL[3]: supervised imitation learning; VPRL[4]: indication of whether using visual prospective representation learning or not; RMSE[5]: root mean square error of steering prediction; Acc.[6]: average accuracy of collision prediction; [7]: Calculated regression accuracy upon corresponding data set, $\text{acc}_\epsilon = \text{count}\left(|y'_{\text{pred}} - y_{\text{true}}| < \epsilon\right)/n$ based on the original reference (Zheng et al. 2022); [8]: Verified based on their provided model with RMSE metrics. *: Retrain on HDIN and DroNet datasets.

+SL. The results of DroNet+SupCR in Table 2 demonstrate that capturing the intrinsic ordered relationship between consecutive samples depending on the SupCR loss function is capable of benefiting the regression performance.

One should notice that the DroNet + SL, ACO + CL, ViViT + CL and DroNet+SupCR have not applied VPRL to generate prospective representations for anticipating future states. In comparison, the SOTA DeepExplorer+SIL generates a visual prospective representation z'_{t+1} and obtains the best performance of steering prediction on the DroNet data set. Moreover, the proposed MulVPRL framework obtains regression-aware ($z'_{reg_{t+1}}$) and classification-aware ($z'_{cls_{t+1}}$) prospective representations as shown in Figure 8f, and consequently improves the steering regression and collision classification. To find out why the MulVPRL framework only obtains the best results on the HDIN data set while the DeepExplorer performs better in steering prediction tasks on the DroNet data set as shown in Table 2, ablation studies have been conducted in the next subsection to investigate reasons.

5.3 | Ablation Study

Figure 9 intuitively presents the overview of structural variants with respect to the proposed MulVPRL and the SOTA DeepExplorer for the ablation study. The structure variants with details will be illustrated below.

MulVPRL_*Enc* (Figure 9a): This structure conjunctly trained the ResNet8 encoder along with projectors and *MLP* layers of steering regression and collision classification. This structure simultaneously predicts steering values and collision probability from the current image x_t.

MulVPRL_*Enc*with*GRU* (Figure 9b): A GRU layer, the same as the GRU layer used by the MulVPRL framework, is added to the MulVPRL_*Enc* for conjunctional training and extracting the current spatial-temporal representation c_t from the latest sequential images between x_{t-n} to x_t. Subsequently, c_t is fed into the projectors and *MLP* layers for steering and collision predictions.

MulVPRL_no*ClsProjClsMLP* (Figure 9c): It is modified from the MulVPRL framework which discards the classification-related projector and *ClsMLP* layer. The spatial-temporal representation c_t purely contains steering-aware features for the regression projector to learn the visual prospective steering-aware representation $z'_{reg_{t+1}}$.

MulVPRL_no*ClsMLP* (Figure 9d): It is modified from the MulVPRL framework which merely discards the classification *ClsMLP* layer. In comparison with MulVPRL_no*ClsProjClsMLP* in Figure 9c, the spatial-temporal representation c_t is partially occupied by collision-aware features.

DeepExplorer_MulVPRL (Figure 9e): The DeepExplorer model is modified to match with the MulVPRL framework. In comparison with the original DeepExplorer in Figure 8e, its ResNet18 encoder is replaced by the ResNet8 encoder that is the same as the MulVPRL_*Enc* structure, and two LSTM layers are replaced by one GRU layer. The sequence length is as same as the MulVPRL framework.

MulVPRL_no*RegProjRegMLP* (Figure 9f): Similar but different to MulVPRL_no*ClsProjClsMLP* from Figure 9c, the MulVPRL framework discards the regression-related projector and *RegMLP* layer. The spatial-temporal representation c_t only contains collision-aware features for the collision projector to learn the visual prospective collision-aware representation $z'_{cls_{t+1}}$.

MulVPRL_no*RegMLP* (Figure 9g): Similar but different to MulVPRL_no*ClsMLP* from Figure 9d, the MulVPRL framework merely discards the regression *RegMLP* layer. Comparing with MulVPRL_no*RegProjRegMLP*, the spatial-temporal representation c_t involves both steering-aware and collision-aware features.

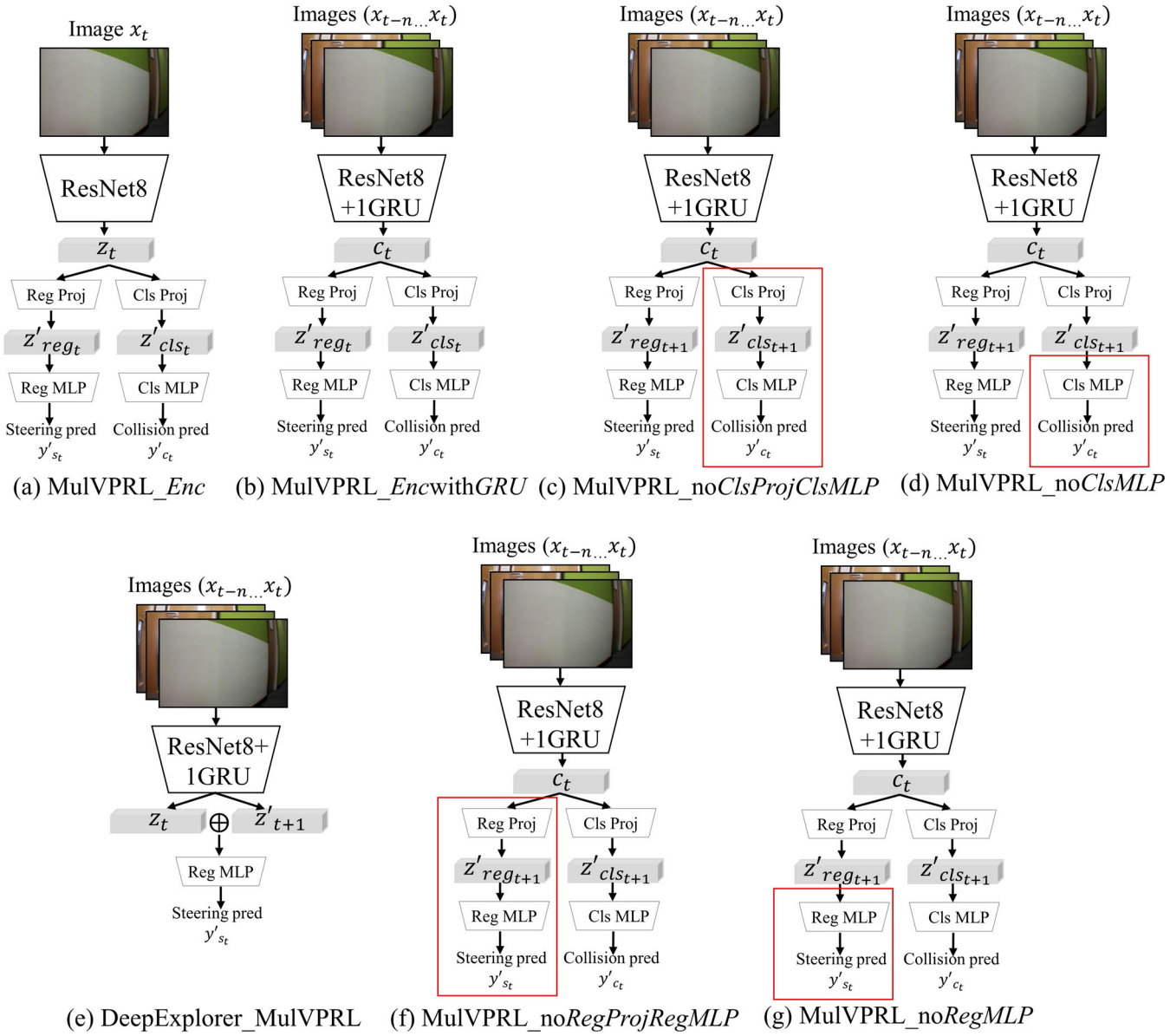


FIGURE 9 | Overview of structural variants for ablation study. The regression and classification projectors within structural variants are shown as *RegProj* and *ClsProj* while the regression and classification *MLP* layers are marked as *RegMLP* and *ClsMLP*. (a) MulVPRL_*Enc* (b) MulVPRL_*Enc*with*GRU*. (c), (d), (f), and (g) represent ablated structures of the original MulVPRL framework by removing the sub-components within the red solid box for comparison. [Color figure can be viewed at wileyonlinelibrary.com]

Table 3 shows the results of the MulVPRL framework and DeepExplorer model with respect to different structural variants and the best results are highlighted in bold in the last four columns. The "Tasks" column indicates that the corresponding models focus on multi-tasks, single regression task or single classification task in which the multi-tasks involves regression task for steering prediction and classification task for collision prediction. The "Loss Functions" column indicates the corresponding loss functions used by the model for different training stages. The structural variants in the "Models" column are illustrated as aforementioned. The "-" in Table 3 means the corresponding structure does not provide this prediction.

The MulVPRL_*Enc* is similar to the baseline DroNet model, where both models employ the same ResNet8 encoder to predict steering values and collision probability by purely using a present image, hence they obtain similar results. The MulVPRL_*Enc*with*GRU* extracts the spatial-temporal representations from the latest sequential images and therefore, further benefits the prediction performance, especially the steering regression performance compared with the the MulVPRL_*Enc*. Moreover, since the proposed MulVPRL framework considers VPRL for regression and classification sub-tasks compared with the MulVPRL_*Enc*with*GRU*, it performs the best with respect to the structures in the "Multi-tasks" row in Table 3.

As for the single regression task, the MulVPRL_no *ClsProjClsMLP* without classification-related components is sorely responsible for the steering regression task. In comparison with the MulVPRL_no*ClsMLP*, the better regression performance of the MulVPRL_no*ClsProjClsMLP* in Table 3 demonstrates that the model purely learns one type of visual

TABLE 3 | Results comparison of MulVPRL and DeepExplorer structural variants.

Tasks	Loss functions [1]	Models	Len [9]	Num. params [10]	Results on HDIN data set		Results on DroNet data set	
					RMSE	Acc.	RMSE	Acc.
Multi-tasks	$\mathcal{L}_{MulSL}$ [2]	**DroNet**	1	0.3M	0.123	85.8%	0.110	95.4%
		MulVPRL_Enc	1	1.6M	0.125	84.0%	0.110	95.0%
		MulVPRL_EncwithGRU	4	1.9M	0.098	**86.7%**	0.104	95.1%
	$\mathcal{L}_{MulVPRL}$ [3]+$\mathcal{L}_{MulSL}$	**ours MulVPRL**	4	2.0M	0.083	86.6%	0.098	96.0%
Single Reg. task	$\mathcal{L}_{SupCR}$ [4]+$\mathcal{L}_{MSE}$ [5]	MulVPRL_noClsProjClsMLP	4	2.0M	**0.081**	—	0.092	—
	$\mathcal{L}_{MulVPRL}$ + $\mathcal{L}_{MSE}$	MulVPRL_noClsMLP	4	2.0M	0.083	—	0.097	—
	SIL [6]	**DeepExplorer**	10	16M	0.085	—	**0.091**	—
		DeepExplorer_MulVPRL	4	2.0M	0.096	—	0.108	—
Single Cls. task	$\mathcal{L}_{SupCon}$ [7]+$\mathcal{L}_{BCE}$ [8]	MulVPRL_no*RegProjRegMLP*	4	2.0M	—	86.6%	—	**97.5%**
	$\mathcal{L}_{MulVPRL}$ + $\mathcal{L}_{BCE}$	MulVPRL_no*RegMLP*	4	2.0M	—	86.6%	—	96.1%

Note: Loss Functions[1]: In this column, "+" separates the loss functions used for the 1st and 2nd training stages while without "+" represents only one training stage; $\mathcal{L}_{MulSL}$ [2]: MulSL loss function from Equation (16); $\mathcal{L}_{MulVPRL}$ [3]: MulVPRL loss function from Equation (15); $\mathcal{L}_{SupCR}$ [4]: SupCR loss function from Equation (11); $\mathcal{L}_{MSE}$ [5]: MSE loss function; SIL [6]: Supervised Imitation Learning; $\mathcal{L}_{SupCon}$ [7]: SupCon loss function from Equation (14); $\mathcal{L}_{BCE}$ [8]: BCE loss function; Len [9]: Sequence length of the latest observed images; Num. Params [10]: The number of total parameters of the model, $M = 1e^6$.

prospective representation performs better for the corresponding subtask. This trend has also been evidenced by comparing the results between the MulVPRL_no*RegProjRegMLP* and MulVPRL_no*RegMLP* in the single classification task of Table 3. In addition, the original DeepExplorer which is also entirely responsible for the regression task merely achieves an improvement of 0.001 RMSE on the DroNet data set compared with the MulVPRL_no*ClsProjClsMLP*, but requires a greater number of parameters (16M *vs*. 2.0M parameters) and longer image sequence (10 vs. 4 sequence length). Finally, suppose the encoder's backbone, *RNN* layer and the sequence length of the DeepExplorer model are replaced by the corresponding components of the MulVPRL framework, that is the DeepExplorer_MulVPRL. In that case, the results are significantly decreased and worse than the proposed MulVPRL framework. According to these comparisons, there are two prime summaries. Firstly, the spatial-temporal representation c_t, which incorporates both steering-aware and collision-aware features, provides less comprehensive information compared to a spatial-temporal representation c_t that is purely focused on either steering-aware or collision-aware features. Secondly, supervised contrastive learning can better enhance the capability of representation learning for smaller-size models compared to supervised imitation learning, and subsequently achieve better performance than larger-size models.

Another observation from Table 3 is that the best accuracy of collision prediction on the HDIN data set is 86.7%. This situation occurred due to the manual annotations for "Collision" and "Non-collision" images shown as the examples in Figure 10. The labels of "Collision" and "Non-collision" images are determined according to the distance between the UAV and the front obstacles, but the wall in Figure 10b is closer to the UAV than that shown in Figure 10a and is labeled as "Non-collision." The model correctly predicts the situation in Figure 10a as $y'_{c_t} = y_{c_t} = 1$ and thus, erroneously predicts the situation shown in Figure 10b as $y'_{c_t} = 1 \neq y_{c_t} = 0$. To verify whether the annotation error affects the capability of collision avoidance, an experiment has been conducted by flying the UAV straight forward to the obstacle. The UAV stopped in front of the obstacle when the MulVPRL framework predicted the collision probability $y'_{c_t} \geq 0.5$, which shows the annotation error does not impact the capability of collision avoidance.

5.4 | Visualization and Statistical Analyses

In this subsection, the baseline DroNet, the recent SOTA DeepExplorer and the proposed MulVPRL along with their structural variants are specifically selected as the representative models, that is the structural variants highlighted in Table 3 "Models" column. The visualizations along with statistical analyses of their performance are presented for discussions.

Statistical analysis can help in studying the robustness of improvements over baseline models. Compared to the basic p-values and confidence intervals, R^2 and Explained Variance Adaptation (*EVA*) are similar and more appropriate for evaluating the fitting performance of regression models. Same as the baseline DroNet (Loquercio et al. 2018), this study adopts *EVA* as an additional statistical metric in Table 4 for assessing regression performance.

$$EVA = 1 - \frac{VAR\left[y_{true} - y_{pred}\right]}{VAR\left[y_{true}\right]}. \tag{17}$$

EVA is calculated as Equation (17) in which *VAR* represents the variance. The greater the *EVA*, the better the alignment

(a) "Collision" image (b) "Non-collision" image

FIGURE 10 | (a) Actual label $y_{c_t} = 1$, model correctly predict $y'_{c_t} = 1$. (b) Actual label $y_{c_t} = 0$, model erroneously predict $y'_{c_t} = 1$. (a) "Collision" image, (b) "Non-collision" image. [Color figure can be viewed at wileyonlinelibrary.com]

TABLE 4 | Metrics and results for statistical analyses.

Models	Results on HDIN data set				Results on DroNet data set			
	RMSE	*EVA*	Acc.	*F*1	RMSE	*EVA*	Acc.	*F*1
DroNet	0.123	0.827	85.8%	0.794	0.110	0.737	95.4%	0.895
DeepExplorer	0.085	0.924	—	—	0.091	0.825	—	—
DeepExplorer_MulVPRL	0.096	0.900	—	—	0.108	0.767	—	—
ours MulVPRL	0.083	0.923	86.6%	0.807	0.098	0.788	96.0%	0.907

between the model's predictions and ground truths concerning the continuous data. The intuitive visualizations are shown in Figure 11a which respectively depicts the chronological steering curves and distributed histograms concerning predictions and ground truths. It can be observed that the steering predictions of DeepExplorer, DeepExplorer_MulVPRL and the proposed MulVPRL framework align more closely with the actual steering curves. The DeepExplorer and the proposed MulVPRL model obtained similar and optimal *EVA* on the HDIN data set, and the DeepExplorer achieved the best *EVA* on the DroNet data set.

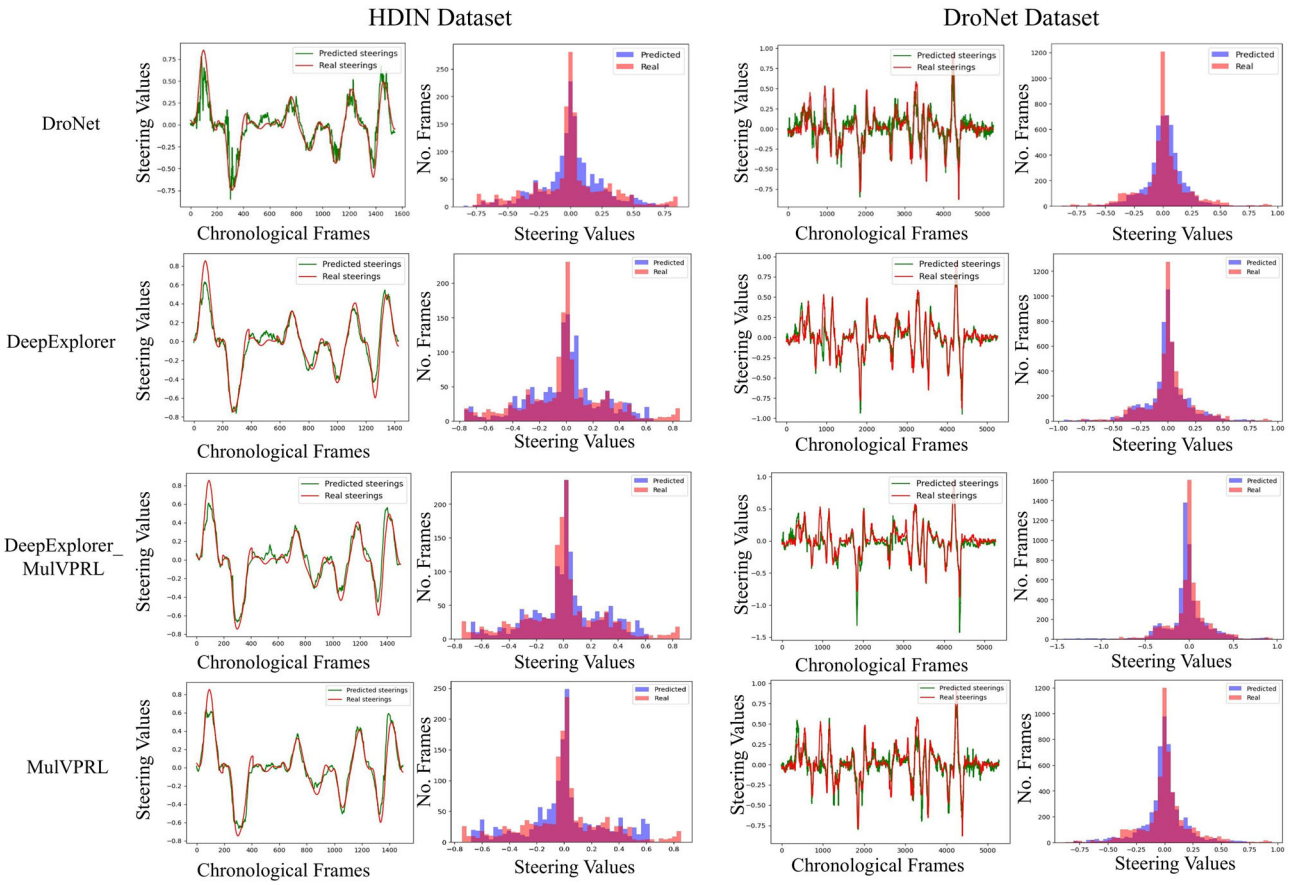
As for collision prediction, the *F*1 score as a statistical metric concerning both precision and recall for imbalanced data is additionally used in Table 4 to indicate the classification performance of the baseline DroNet and the proposed MulVPRL framework. For intuitive understanding, the confusion matrices of DroNet and MulVPRL framework are shown in Figure 11b. It can be observed that these two models have correctly predicted all "Collision" images on the HDIN data set while a small number of "Non-collision" images are erroneously misclassified. These false positive cases are consistent with situations shown in Figure 10 to corroborate the labeling errors of collision-related images. Since all "Collision" images are correctly classified, the MulVPRL framework with better *F*1 scores will improve the capability of collision avoidance for UAV control, that is "Non-collision" prediction ($y'_{c_t} < 0.5$) allows to go forward while "Collision" prediction ($y'_{c_t} \geq 0.5$) requires to stop.

In addition, the Umap (McInnes et al. 2018) and t-SNE (van der Maaten and Hinton 2008) visualization technologies are respectively used to present the distributions of regression-aware representations shown in Figure 12a and that of classification-aware representations shown in Figure 12b. These colored representation distributions can help to intuitively understand how the model segregates the visual representations.

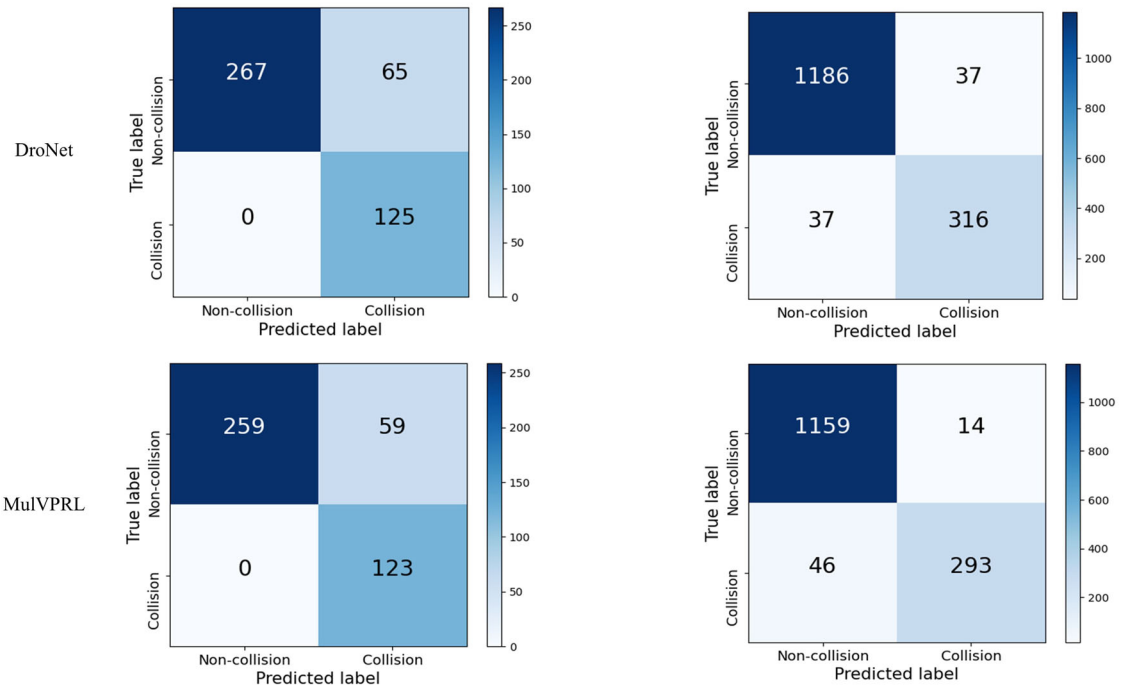
As shown in the final row of Figure 12a, the MulVPRL framework captures the intrinsic ordered relationship between consecutive visual prospective regression-aware representations, and the representation distributions are similar to that in Figure 4. In comparison, the baseline DroNet model without VPRL obtains the unsorted present regression-aware representations which are distributed in the 1st row of Figure 12a.

The DeepExplorer model based on supervised imitation learning is also dedicated to learn the visual prospective regression-aware representation. It achieves an improvement of 0.007 RMSE on the DroNet data set compared to the MulVPRL framework in Table 3. Through observing the second row in Figure 12a, the visual prospective regression-aware representation of the DeepExplorer model presents a sorted distribution trend that is similar to the MulVPRL framework, but requires 8 times the number of parameters.

If the DeepExplorer model adopts the same components as the MulVPRL framework but maintains its original supervised imitation learning method, as shown in the third row of Figure 12a, the DeepExplorer_MulVPRL with fewer parameters would struggle to sort the extracted visual prospective regression-aware representation following the order of label values. Comparing the representation distributions between the DeepExplorer_MulVPRL and the proposed MulVPRL framework will remove the impact of different model structures



(a) Steering Curves and Histograms



(b) Collision Confusion Matrices

FIGURE 11 | Visualizations of steering and collision predictions. (a) presents steering curves and distributed histograms concerning predictions and ground truths. Green curves and blue histograms are predicted results while red curves and orange histograms are ground truths. (b) shows the confusion matrices of prediction. [Color figure can be viewed at wileyonlinelibrary.com]

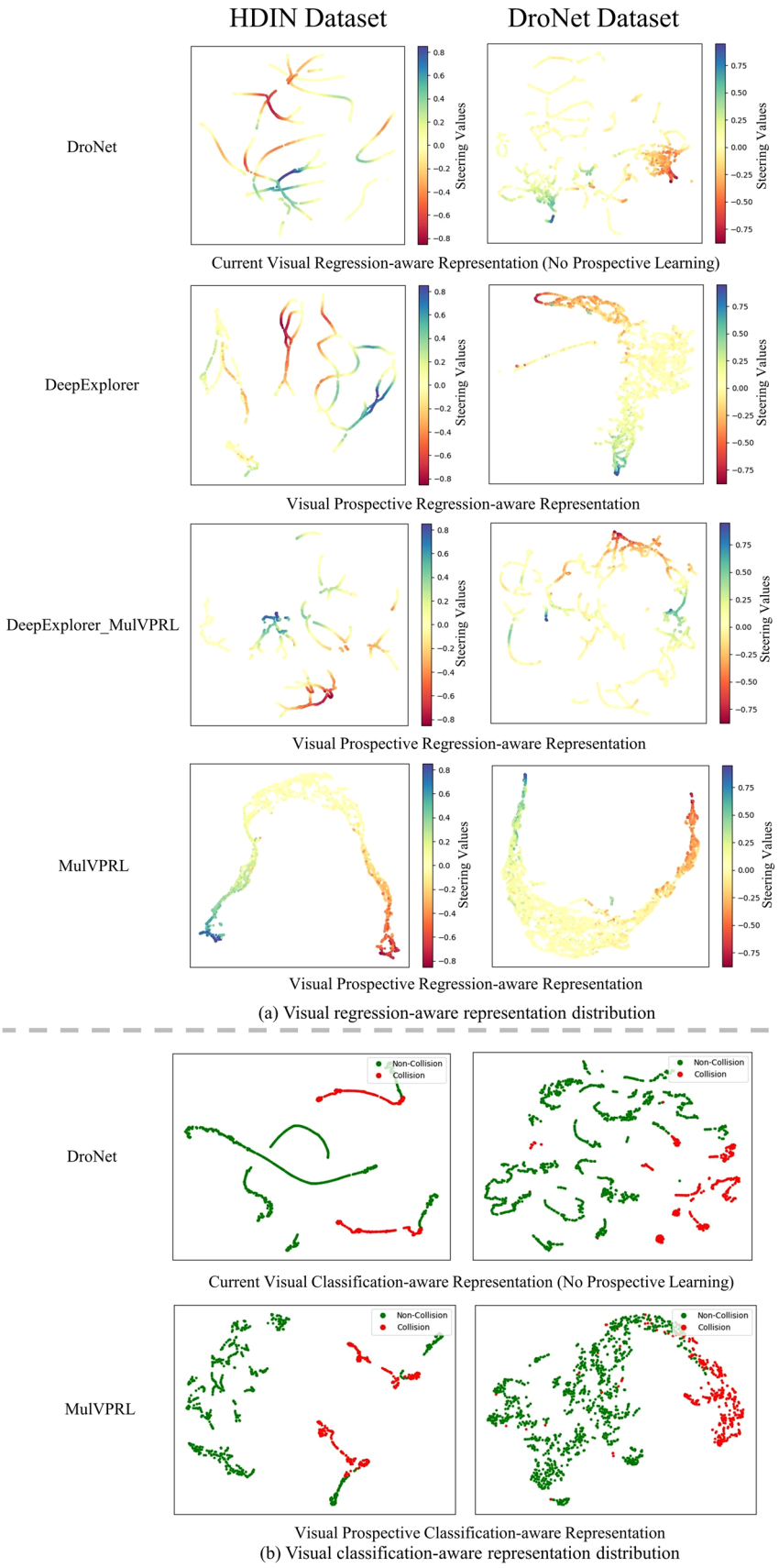


FIGURE 12 | Legend on next page.

and aim to only analyze the performance of obtaining visual prospective regression-aware representations based on different learning methods. In conclusion, the supervised contrastive regression of the MulVPRL framework performs better than the supervised imitation learning of DeepExplorer_MulVPRL for sorting the representation in an order and thereby, benefits steering prediction.

Figure 12b presents the classification-aware representation distributions of the multi-task DroNet model and MulVPRL framework. According to the observations, the visual prospective classification-aware representations obtained from the MulVPRL framework perform clearer segregation. Therefore, it is capable of benefiting the following MLP layers to be trained during the second training stage described in Section 4.2 with shorter training procedures and better classification performance.

6 | Real-World Experiments

This section will conduct real-world experiments in unknown indoor environments using an actual UAV platform. The same compared models in Section 5.4 are selected, that is the baseline DroNet, SOTA DeepExplorer, DeepExplorer_MulVPRL and the proposed MulVPRL framework, and trained on the indoor HDIN data set. Section 6.1 will introduce the experiment setup while a pretest is conducted for data augmentation in Section 6.2. The entire navigation experiments are visually and quantitatively shown in Section 6.3. Finally, supplementary experiments and additional discussions concerning diverse environmental conditions and agents are involved in Section 6.4.

6.1 | Experiment Setup

This section will sequentially introduce practical environments, UAV platform and the control policy.

Environments

Figure 13 shows the top-down views of real-world practical environments. These environments without junctions will be mainly used to evaluate the performance of short-term reactive control policies. Notably, the narrowest width of these corridors is 1.4 m, and the UAV will be operated at a constant altitude of 0.7 m.

UAV platform

Due to the indoor environments requiring the UAVs to be capable of vertically taking off and landing, and hovering in the same location, the rotorcraft UAVs are more suitable than other UAV platforms. In recent years, there have been many commercial and open-source supported quadcopters such as Parrot AR. Drone (Parrot 2012) and Crazyflie (Bitcraze 2019), which facilitate scientific research.

Furthermore, Palossi et al. (Palossi et al. 2019) classified the rotorcraft UAVs into four different groups according to their sizes and payloads in Table 5. According to this classification, the Parrot AR. Drone 2.0 shown as Figure 14a with around 60 cm radius can be treated as a Standard-size UAV while the Crazyflie 2.1 shown as Figure 14b with a 9.2 cm radius is regarded as a Nano-size UAV.

Although both Parrot AR. Drone 2.0 and Crazyflie 2.1 match the physical constraints of the narrowest corridor, that is the UAV's size should be smaller than Width = 1.4 m in environments of Figure 13a,b, the Parrot AR. Drone 2.0 cannot stably hover in this place due to the wind disturbance caused by its larger size and powerful propellers. Due to the summarized main reasons below, the Nano-size Crazyflie 2.1 is chosen as the experimental UAV platform. This platform equips with a 320 × 240 greyscale monocular camera and the memory size is 8M *bytes*.

- UAV's stability. All models for real-world experiments focus on locomotion control rather than stabilization control.
- Models' generalization capability. Images of the HDIN data set were collected from Parrot AR. Drone 2.0 for training models and should be validated on a different platform.

Control Policy

The DroNet, DeepExplorer, DeepExplorer_MulVPRL and the proposed MulVPRL framework process a short-term image sequence and predict the steering value y'_{s_t} and collision probability y'_{c_t} for reactive control. y'_{s_t} represents the ratio of maximum angular velocity ranging in $[-1, 1]$ while y'_{c_t} means the probability of collision within $[0, 1]$. Since the Crazyflie UAV is controlled by the actual steering angular and linear forward velocities such as $\theta_t = 10°/s$ and $v_{x_t} = 0.5m/s$, the predicted steering and collision values cannot be directly used. Therefore, the following will demonstrate the command transformation of the proposed MulVPRL framework as an example.

Based on the low-pass filter which is inspired by the original DroNet references (Loquercio et al. 2018; Palossi et al. 2019) shown as Equation (18), the collision probability y'_{c_t} $(0 \le y'_{c_t} \le 1)$ is converted to the linear forward velocity v_{x_t} $(0 \le v_{x_t} \le V_{max})$ by multiplying the maximum velocity V_{max} as $(1 - y'_{c_t})V_{max}$. That is, a greater collision probability results to a smaller linear forward velocity. Also, the linear forward velocity v_{x_t} is smoothed by taking the factor α and concerning the

FIGURE 12 | Visualization of representation distribution. Each row represents a corresponding model while the columns respectively present the visualizations of representation distributions on the HDIN and DroNet datasets. Each dot in sub-figures represents an image's representation while the color indicates the ground truth of steering values and collision labels. Specifically, the colors of dots in (a) from red warm → blue cold correspond to the values of steering labels from low → high. The green and red dots in (b), respectively, denote the 'Non-collision' and 'Collision' image labels. The proposed MulVPRL framework in (a) sorts regression-aware representations according to steering values, and better segregates classification-aware representations in (b). [Color figure can be viewed at wileyonlinelibrary.com]

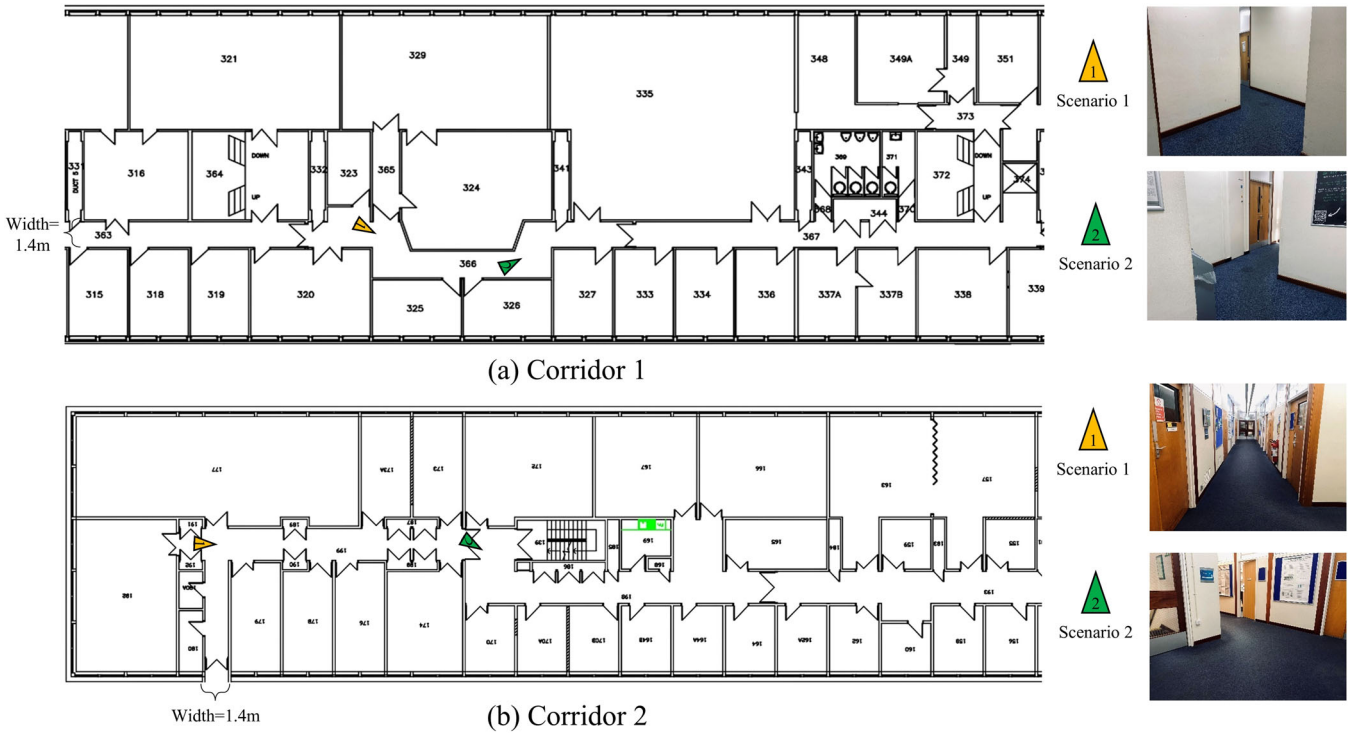


FIGURE 13 | Two real-world unknown corridors. (a) Corridor 1 consists of straight-forward and S-shape segments while (b) Corridor 2 includes straight-forward, L-shape and S-shape segments. The key positions with different orientations are denoted as a colored triangle with a number within environments, and their internal views are taken and presented on the right. [Color figure can be viewed at wileyonlinelibrary.com]

TABLE 5 | Rotorcraft UAV classification.

Vehicle Class	Size	Payload
Standard-size	50 cm	≥1 kg
Micro-size	25 cm	0.5 kg
Nano-size	10 cm	0.01 kg
Pico-size	2 cm	0.001 kg

previous linear forward velocity $v_{x_{t-1}}$. The default $\alpha = 0.2$ and $V_{max} = 0.2$ m/s.

$$v_{x_t} = (1-\alpha)v_{x_{t-1}} + \alpha\left(1 - y'_{c_t}\right)V_{max}. \tag{18}$$

The steering labels of the HDIN data set used for training represent the ratios of maximum angular velocity within [-1, 1] since the original angular velocities are divided by the default maximum angular velocity $S_{max} = 40°/s$. As a result, Equation (19) directly transforms the predicted steering value y'_{s_t} $(-1 \le y'_{s_t} \le 1)$ back to the actual steering angular velocity θ_t $(-S_{max} \le \theta_t \le S_{max})$ by multiplying the same S_{max} value without fine-tuning.

$$\theta_t = y'_{s_t} S_{max}. \tag{19}$$

Finally, the proposed MulVPRL framework can be used to control the UAV by processing the latest 4 greyscaled images between I_{t-3} and I_t for generating the steering angular θ_t and linear forward velocities v_{x_t} as described in Algorithm 2. As for other models, the computations of the *LowPassFilter* and the *Transformation* respectively concerning collision and steering predictions are the same as the MulVPRL framework according to Equations (18) and (19).

Algorithm 2 MulVPRL framework for UAV control

Input: Latest 4 greyscaled images $[I_{t-3}, ..., I_t]$

Output: UAV commands $[v_{x_t}, \theta_t]$

Initialized $\alpha = 0.2$, $V_{max} = 0.2m/s$, $S_{max} = 40°/s$;

while *iteration* **do**

$[y'_{c_t}, y'_{s_t}] = MulVPRL([I_{t-3}, ..., I_t])$;

$v_{x_t} \leftarrow LowPassFilter(v_{x_{t-1}}, \alpha, y'_{c_t}, V_{max})$ according to Equation (18);

$\theta_t \leftarrow Transformation(y'_{s_t}, S_{max})$ according to Equation (19);

Output $[v_{x_t}, \theta_t]$;

end while

Specifically, the DroNet model predicts the steering and collision values via the present image and will be respectively transformed to steering angular and linear forward velocities by following the procedures in Algorithm 2. In addition, the DeepExplorer and DeepExplorer_MulVPRL also follow the procedures in Algorithm 2 by respectively processing 10 and 4 sequential images, but they do not have collision prediction y'_{c_t} to be transformed to the linear forward velocity v_{x_t}. Therefore,

(a) Parrot AR. Drone 2.0 (b) Crazyflie 2.1

FIGURE 14 | Examples of UAV platforms. (a) Parrot AR. Drone 2.0 (b) Crazyflie 2.1. [Color figure can be viewed at wileyonlinelibrary.com]

the collision prediction from the proposed MulVPRL framework is combined with them, that is simultaneously executing the MulVPRL framework for obtaining the linear forward velocity v_{x_t} via the predicted y'_{c_t} based on Equation (18). Since the DeepExplorer, DeepExplorer_MulVPRL and the proposed MulVPRL framework apply the same collision avoidance scheme, the comparisons are fair.

6.2 | Pretest for Data Augmentation

The proposed MulVPRL framework trained on the original HDIN data set without data augmentation is pretested in a common L-shape corridor where the width of the corridor is $1.4m$. The left and right steering trajectories are shown as the red dashed lines in Figure 15a,b by executing Algorithm 2, where the parameters of Equations (18) and (19) are set as $\alpha = 0.2$, $V_{max} = 0.2$ m/s and $S_{max} = 40°/s$.

The UAV exhibits an overcompensation behavior when the UAV makes left steering when approaching the bending center. As the red dash line shown in Figure 15a. However, when the UAV approaches the bending center in the opposite direction to steer right, the trajectory is smooth and continuous as the red dashed line shown in Figure 15b. To investigate this abnormal behavior, that is over-turning to the right, the distributions of actual and predicted steering values have been depicted as a histogram in Figure 15c. It can be observed that the distribution of actual steering values from the original HDIN data set is imbalanced, particularly the number of right steering values (i.e., negative values) is greater than that of left steering values (i.e., positive values). This imbalanced data distribution can lead to imbalanced steering prediction and right-side overcompensation.

To address this issue, data augmentation was used by horizontal flipping of all images in the HDIN data set including the "Steering" and "Collision" subsets to create a horizontally mirrored version of the HDIN data set. Since images on each trajectory follow the temporal sequence, this horizontal flipping does not change the image's order. By combining the mirrored and the original HDIN datasets, the number of trajectories containing ordered images is expanded, and the numerical distribution of steering labels is balanced as shown in Figure 15d. The MulVPRL framework was retrained with data augmentation and retested at the same L-shape corner, the trajectories were plotted as green solid lines in Figure 15a,b. The right-side overcompensation disappears and the trajectories become smooth and continuous.

As for fair comparison, the DroNet, DeepExplorer and DeepExplorer_MulVPRL models that will be used to compare with the proposed MulVPRL framework for analyzing the performance in real-world environments are also retrained with data augmentation by horizontal flipping on the HDIN data set. The results are shown in Table 6.

Since the embedded micro-controller on the Crazyflie (Bitcraze 2019), that is the AI-deck (Bitcraze 2023), only supports 8-bit data, requiring model quantization which is not the focus of this study, all models are executed on the Intel i7-4600U CPU of the laptop and sends the linear forward and angular velocities to the UAV. Therefore, Table 6 used to present computational efficiency involves the models' inference time [*fps*] which might affect the real-time performance. Moreover, the memory requirements (Reqs.) and limitations (Lim.) for onboard processing are shown in the final column of Table 6. It can be observed that HyperRAM for real-time responses has a 8.0M *bytes* memory limitation and thus, only the DroNet and MulVPRL models with light-weight architectures (1.3 M and 7.8 M *bytes*) can be deployed onboard as short-term reactive control policies in future works.

6.3 | Performance in Corridors

The parameters in Equations (18) and (19) for converting the predictions to actual steering angular and linear forward velocities are set as $\alpha = 0.2$ and $S_{max} = 40°/s$ for both corridors in Figure 13. The maximum forward velocity $V_{max} = 0.2$m/s is used for experiments in Corridor 1 and changed to $V_{max} = 0.3$m/s for experiments in Corridor 2. Different maximum forward velocities can assist in analyzing the control precision and reaction speed in varying environments.

The UAV takes off at the same position respectively in two corridors. The models automatically control the UAV without human intervention once the mission starts except in situations when collision, being trapped at a corner or reaching the end of the corridor. A "Landing" command from a human inspector will be sent under these situations.

The DroNet model performs similar trajectories to the MulVPRL framework, where both models successfully control the UAV for reaching the end of Corridor 1 shown as the first and fourth and trajectories in Figure 16a. Although the DroNet model performs the worst in terms of steering and collision predictions on the HDIN data set in Table 6, its light-weight

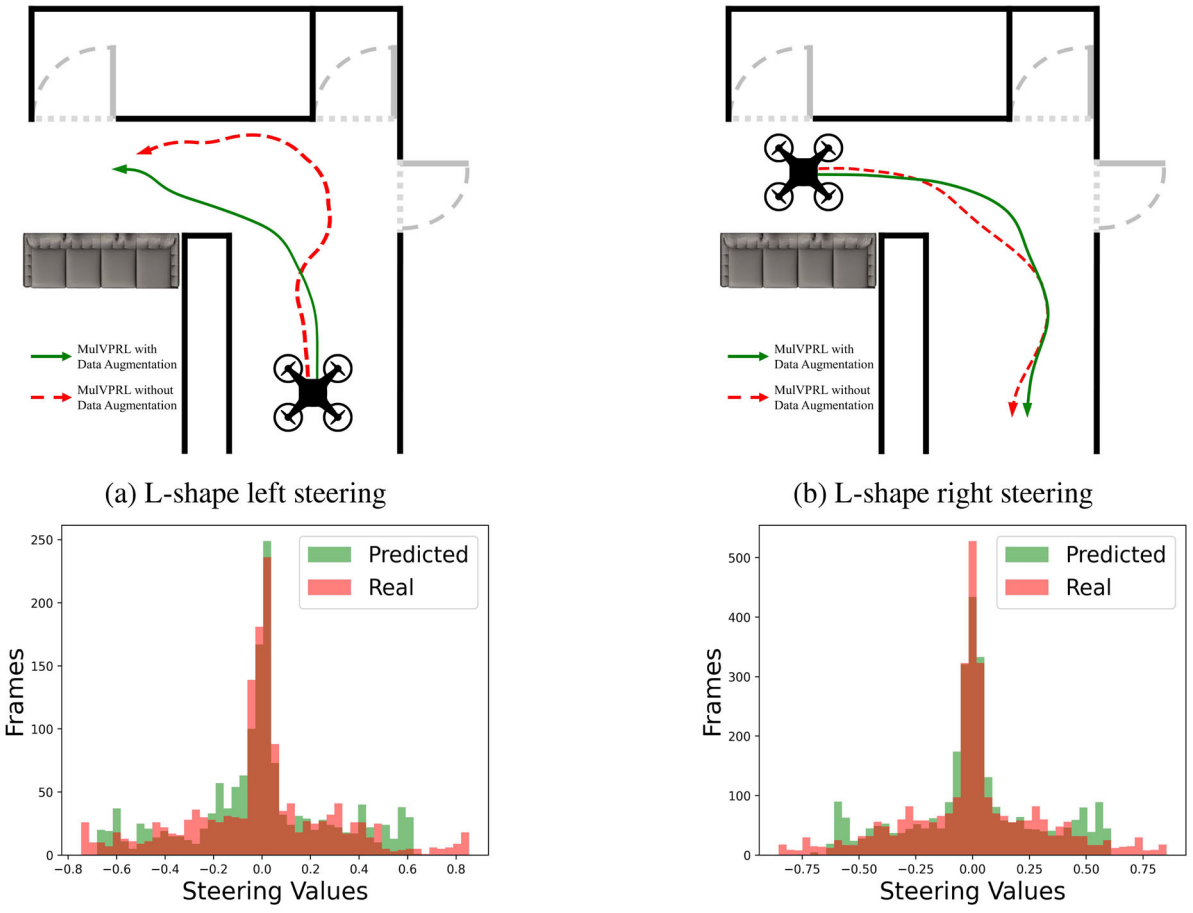


FIGURE 15 | (a) L-shape left and (b) right steering trajectories with and without data augmentation. The green solid trajectory is smoother than the red dashed trajectory in (a) since the model trained with data augmentation. (c) and (d) show the distributed histograms of predicted and actual steering values with and without data augmentation. The data distribution in (d) is more balanced than that in (c). (a) L-shape left steering, (b) L-shape right steering, (c) MulVPRL model training without data augmentation, (d) MulVPRL model training with data augmentation. [Color figure can be viewed at wileyonlinelibrary.com]

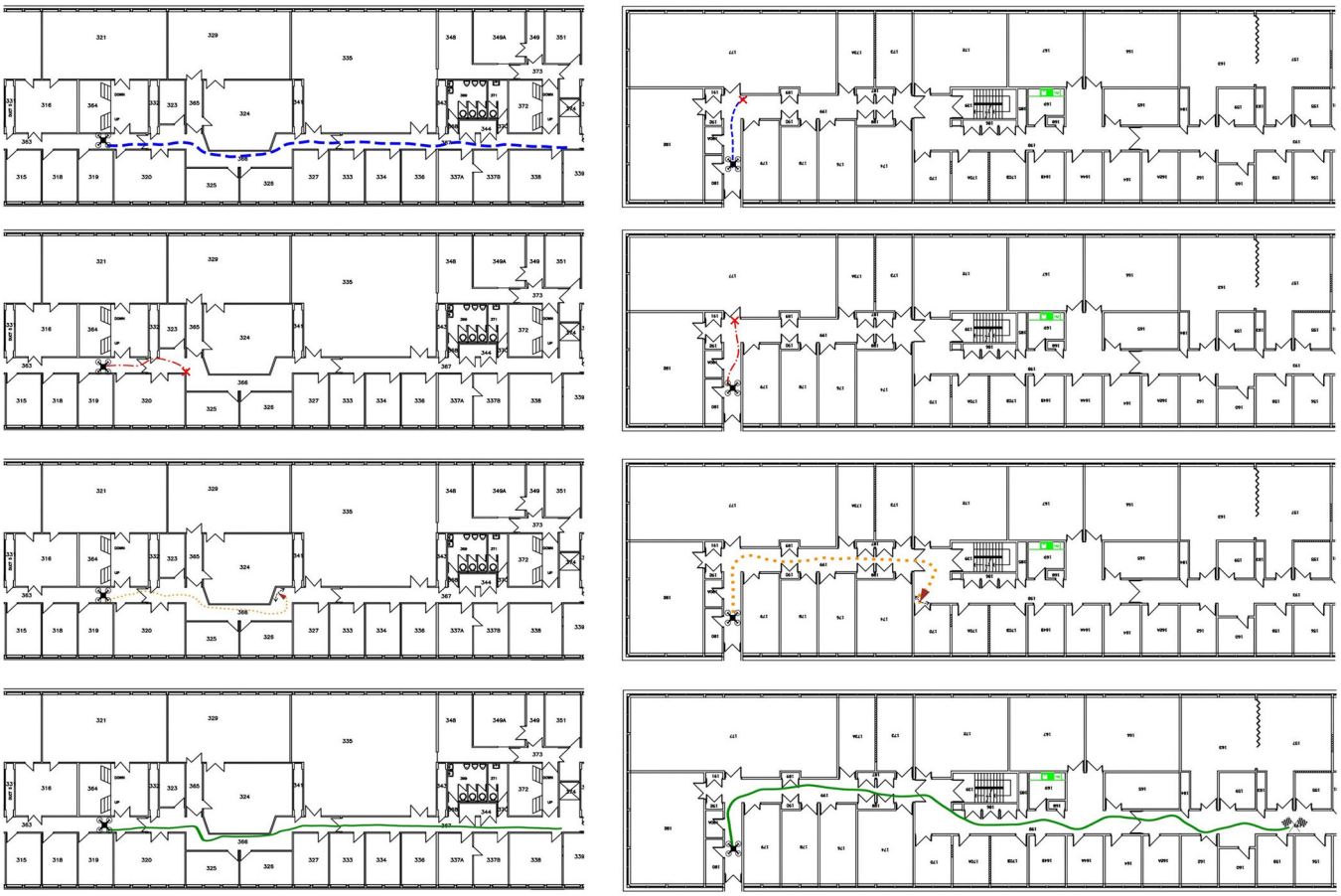
TABLE 6 | Results of models trained with flipping data augmentation along with hardware computational efficiency.

Models	Results on HDIN data set		Inference speed [*fps*]	Reqs./Lim.[1] [*bytes*]
	RMSE	Acc.		
DroNet	0.120	82.9%	5.5	1.3 M/8.0 M
DeepExplorer	0.094	—	1.8	107.0 M/8.0 M
DeepExplorer_MulVPRL	0.100	—	4.3	14.1 M/8.0 M
ours MulVPRL	0.086	84.1%	5.0	7.8 M/8.0 M

Note: Reqs. /Lim.[1]Models' memory requirements and the onboard HyperRAM limitation.

architecture (1.3 M *bytes*) enables rapid orientation adjustment when navigating under a lower maximum UAV velocity (0.2 m/s). In addition, Corridor 1 only involves straight-forward and S-shape corridors with small-angle steering which are less than 90° and hence, the DroNet model can navigate along this corridor without collision. However, due to the limited steering and collision predictions and faster forward velocity (0.3 m/s), the DroNet model exhibits insufficient steering and braking at the large-angle L-shape corner with 90° in Corridor 2, and a collision occurs shown in the first trajectory in Figure 16b.

The DeepExplorer obtains better steering prediction than the DroNet model on the HDIN data set shown in Table 6, and also utilizes the collision prediction from the MulVPRL framework for collision avoidance. However, it still collides on the wall at the entrance of the first turn shown as trajectories of the second row in Figure 16a,b. The major reason lies on its architecture with the greatest size (107.0 M *bytes*) that results in the slowest reaction speed (1.8 fps) for orientation adjustment and deceleration.



(a) Trajectories in Corridor 1 (b) Trajectories in Corridor 2

FIGURE 16 | The UAV takes off at . and respectively represents collision situation and being trapped at a corner. denotes that the UAV reaches the end of corridors. Blue dashed lines, red dashed-dotted lines, yellow dotted lines and green solid lines respectively indicate the trajectories of DroNet, DeepExplorer, DeepExplorer_MulVPRL and MulVPRL models. (a) Trajectories in Corridor 1 (b) Trajectories in Corridor 2. [Color figure can be viewed at wileyonlinelibrary.com]

Furthermore, the DeepExplorer_MulVPRL reduces the number of parameters compared to the DeepExplorer and thereby improves inference speed (14.1 M *bytes*, 4.3*fpsvs*. 107.0M *bytes*, 1.8 fps). Though the DeepExplorer_MulVPRL model does not obtain better steering prediction than the DeepExplorer model in Table 6, it is able to autonomously fly for longer trajectories in both corridors shown in the third row of Figure 16a,b. Nevertheless, since the DeepExplorer_MulVPRL model exhibits similar errors of unexpected steering prediction at the exits of the S-shaped corners in both corridors, the UAV is trapped at the corner and cannot escape. Being trapped at the corner is related to the reactive policy that only observes short-term information and hence, exhibits a constant left-right swaying situation.

Finally, as the short-term reactive control policy, the proposed MulVPRL framework outperformed other compared models, which not just obtained the best steering and collision predictions on the HDIN data set in Table 6, but also achieved the farthest trajectories with 42.7 m in Corridor 1 and 46.9 m in Corridor 2 by respectively using 227 and 177 s for successfully reaching the end of two corridors. Trajectories are shown in the final row in Figure 16a,b. Table 7 shows the quantitative performance of these four models that are experimented in two corridors including the trajectory length and flight time.

6.4 | Supplementary Experiments and Discussions

Although the primary focus of this paper is on UAV reactive control and navigation, the control strategy illustrated in Section 6.1 incorporates linear forward and steering angular velocities (v_{x_t} and θ_t), which allows the MulVPRL framework to control different types of agents such as the UGVs for movement on a 2D plane. That is, the practical agent for the specific application is not just limited to the UAV. Moreover, the supervised contrastive learning of the MulVPRL framework can be further developed to contrast visual samples during 3D motions by introducing more commands such as vertical velocity v_{z_t}, to consequently facilitate 3D control performance of UAVs in future works.

Moreover, two extra experiments for the MulVPRL framework are conducted without any human intervention once the mission starts except in emergencies when the UAV collides with

TABLE 7 | Real-world experimental performance.

Models	Corridor 1		Corridor 2	
	Traj Len [1]	Time [2]	Traj Len	Time
DroNet	45.9 m	229 s	3.3 m	16.3 s
DeepExplorer	7.51 m	37.6 s	3.8 m	19.1 s
DeepExplorer_MulVPRL	27.3 m	136 s	17.1 m	85.6 s
MulVPRL	42.7 m	227 s	46.9 m	177 s

Note: Abbreviations: Traj Len[1]: UAV's navigation path length; Time[2]: UAV's navigation time.

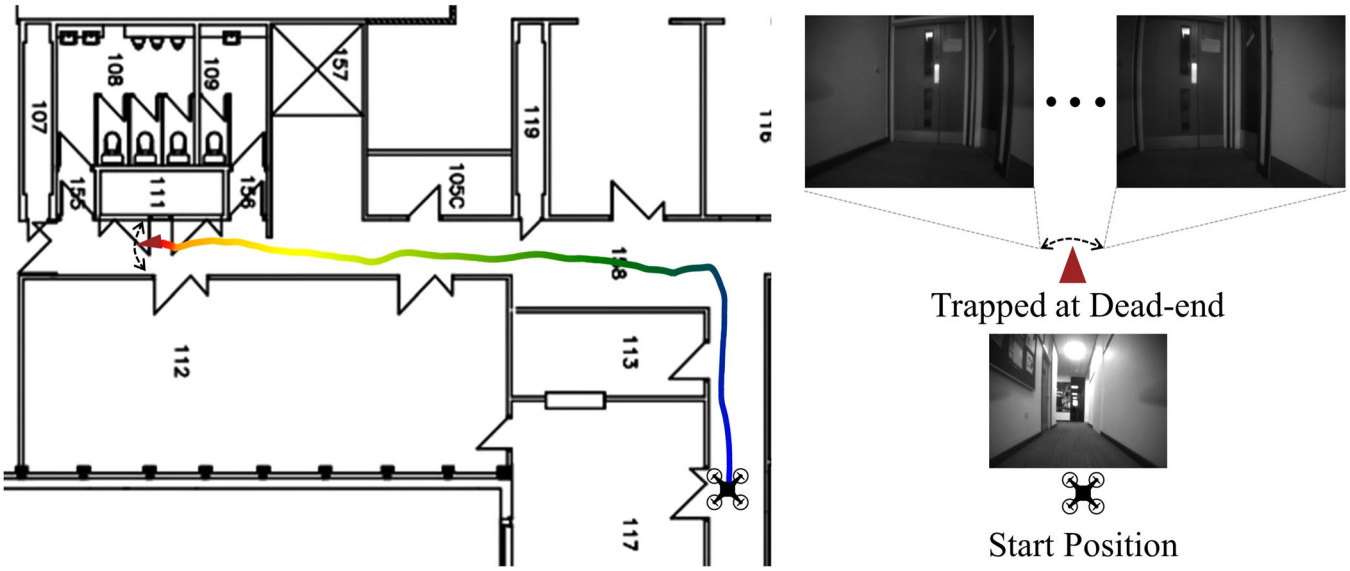


FIGURE 17 | Trajectory of MulVPRL trapped at the dead-end region. The trajectory color from blue cold → red warm indicates the movement's chronological order, and the corridor width is 1.4 m. The UAV takes off at and eventually gets trapped at until it runs out of power. The right figures present the UAV's first-person views at the start and dead-end positions. [Color figure can be viewed at wileyonlinelibrary.com]

obstacles or runs out of power. Specifically, the first experiment is conducted in an unknown corridor with dead-end regions while the second experiment is tested in an enclosed unknown office with randomly placed items such as chairs, tables, clutters, and umbrella. The UAV control procedures follow Algorithm 2 with the parameters of $\alpha = 0.2$, $V_{max} = 0.2$ m/s and $S_{max} = 40°/s$ for Equations (18) and (19) to control the UAV, which is the same parameters of experiments in Corridor 1 in Section 6.3.

The first experiment is shown in Figure 17 in which the UAV turns left at the first junction after taking off. This left turn is sorely reactive control rather than selecting the best route via long-term spatial-temporal observations since the traversable space on the left side is greater than that on the right when entering the junction. Subsequently, the UAV navigates to the dead-end region of the corridor and constantly sways left and right until running out of power. In general, the UAV successfully flies for 23.05 m in this corridor.

Since the MulVPRL framework, as a short-term reactive control policy, is incapable of tackling route-selection at junctions and dead-end returning for long-term tasks in large-scale environments, current solutions include conventional map-building and short-term, long-term integrated methods. As introductions in Section 2.4, conventional SLAM-based map-building algorithms can handle long-term large-scale tasks, but they have massive computational and memory burdens. Moreover, the current research trend is to integrate the short-term reactive policy with a long-term policy such as CNN-based map prediction, and image retrieval and localization. Specifically, these integrated methods achieve long-term tasks in large-scale, even dynamical environments, which not just require precise and instantaneous reactive control commands, but also invoke the long-term policy with updated global information at a constant interval (e.g., 25 steps Chaplot et al. 2020; Ramakrishnan et al. 2020).

The second experiment is illustrated in Figure 18 along with internal views and chronological trajectory. This office size is 6.0 × 4.6 m. Unlike navigating in open spaces such as corridors shown in Figure 13 where the observed images contain distant visual backgrounds (i.e., depth of field) to enable the model to perceive farther navigable spaces, the MulVPRL framework controls the UAV flying in the enclosed space is more like a random navigation with a collision avoidance scheme. The UAV traveled a distance of 15.66 m from the start position by using 123 s. However, although the UAV completed orientation adjustment for collision avoidance at the end of the trajectory which is shown as the top image in Figure 18, the propeller-

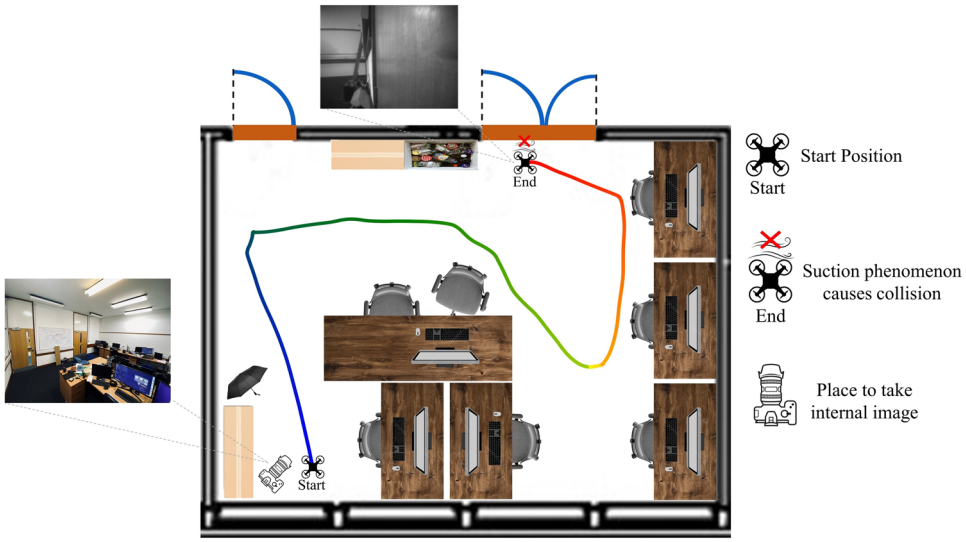


FIGURE 18 | Trajectory of MulVPRL in office. The colored trajectory indicates chronological order. The UAV flies at a constant altitude of 0.7 m, below the height of the desktop, and eventually collides on the wall due to the suction phenomenon caused by Bernoulli's principle. [Color figure can be viewed at wileyonlinelibrary.com]

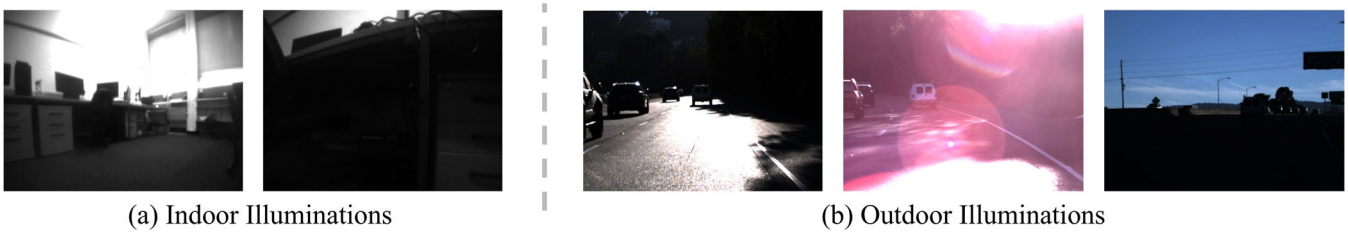


FIGURE 19 | (a) Various lighting conditions in the office. (b) Lighting conditions on vehicle roads. [Color figure can be viewed at wileyonlinelibrary.com]

generated airflow caused by proximity to walls and clutters results in a suction phenomenon according to Bernoulli's principle and thereby, the UAV's side was drawn to the wall and occurs collision. In general, the MulVPRL framework can control the UAV in the enclosed space, but the clutter within narrow environments affects the UAV to be controlled as stability as in open corridors that are free of items.

More environmental conditions are presented as the example images in Figure 19. As for varying lighting conditions, during the experiments in the office, the MulVPRL framework normally controlled the UAV under normal illumination and did not move forward into the dark areas, which are shown in Figure 19a. In addition, the MulVPRL framework was also tested on the outdoor DroNet data set and outperformed the baseline DroNet model presented in Table 2. This data set involves scenarios on vehicle roads under various lighting conditions shown in Figure 19b.

Furthermore, the DroNet model trained on its data set with outdoor image samples (Figure 19b) has demonstrated the capability of controlling the UAV navigating along the outdoor vehicle roads based on the "line-like" patterns (Loquercio et al. 2018). The proposed MulVPRL framework with better performance on the outdoor DroNet data set shown in Table 2 provides the potential to be used in outdoor environments. Since the application objective of this study is indoor navigation and the MulVPRL framework was trained solely on the indoor HDIN data set, future works will be extended to outdoor navigation.

Finally, according to all the real-world experiments in this section, Table 8 summarizes the strengths and weaknesses of the models involved in the comparison by illustrating differences between each other to emphasize the prime reasons.

7 | Conclusion

As an indispensable component of indoor mapless navigation systems, the reactive control policy is expected to feature a light-weight architecture for pursuing instantaneous responsiveness and high control precision. However, the recent models based on different learning methods, such as supervised and contrastive learning, have encountered a bottleneck in advancing prediction performance. Inspired by the "Mirror Flower, Water Moon" idea to anticipate future states for enhancing the model's performance, this paper proposes a novel light-weight MulVPRL framework. To the best of our knowledge, this is the first attempt to integrate the VPRL with supervised contrastive learning, which outperforms other models whether employing VPRL or not and based on different learning methods validating on datasets and real-world environments.

TABLE 8 | Summary table for models in real-world experiments.

Models	Advantages	Disadvantages
DroNet	Light-weight architecture allows fast inference speed for rapid reactions.	Limited prediction performance leads to insufficient steering and deceleration.
DeepExplorer	Large model with VPRL based on SIL initially captures the intrinsic ordered relationship of regression samples and thus benefits prediction on datasets.	Large architectural size delays reaction.
DeepExplorer_MulVPRL	Light-weight architecture improves inference speed compared to DeepExplorer.	Light-weight model with VPRL based on SIL fails to capture intrinsic ordered relationship and hence limits prediction performance.
ours MulVPRL	Light-weight architecture with VPRL based on MulVPRL loss function obtains optimal prediction performance on both datasets and real-world environments.	Short-term reactive control cannot tackle junctions and trapped situations, which are the common disavantages of all short-term reactive policies.

Although the MulVPRL framework has shown competitive performance on reactive control for indoor mapless navigation, it sorely depends on short-term observations while regardless of long-term historical information for large-scale environmental tasks. This results in challenges of handling route-selection at junctions and returning from dead-end regions. Inspired by the short-term, long-term integrated approaches (Chaplot et al. 2020; Ramakrishnan et al. 2020), a global policy within a hierarchical architecture, such as image retrieval (Wei et al. 2024; Leyva-Vallina et al. 2024) for key scenario recognition (Arandjelovic et al. 2016; Liang et al. 2022; Li et al. 2023), can be invoked at a constant interval to be aware of global environmental information. As a result, our future works will be dedicated to exploring long-term mapless control policy and to integrating with our proposed MulVPRL framework, consequently to accomplish tasks such as victim search and rescue in large-scale environments.

In addition, the MulVPRL framework not just has the potential to be used in outdoor environments even on different ground vehicles if trained with data captured from outdoor and associated practical agents, but also found that the model with greater size and parameters is capable of advancing prediction performance. According to the real-time requirements for high-frequency agent control, our future works will develop the advanced and larger MulVPRL framework with powerful encoders based on model distillation and quantization, and deploy it to agents' onboard processors for applying in both indoor and outdoor dynamic environments.

Acknowledgments

The authors acknowledge the assistance from the team member, Kang Xiang and Chi Jen Li, for data collection and video recording. The work has been supported in part by China Scholarship Council (202008010003).

Data Availability Statement

The data that support the findings of this study are available from the corresponding author upon reasonable request.

References

Arandjelovic, R., P. Gronat, A. Torii, T. Pajdla, and J. Sivic. 2016. " Netvlad: CNN Architecture for Weakly Supervised Place Recognition." In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 5297–5307.

Arnab, A., M. Dehghani, G. Heigold, C. Sun, M. Lučić, and C. Schmid. 2021. " Vivit: A Video Vision Transformer." In IEEE International Conference on Computer Vision (ICCV), 6836–6846.

Balntas, V., S. Li, and V. A. Prisacariu. 2018. Relocnet: Continuous Metric Learning Relocalisation Using Neural Nets. pp. 751–767.

Batinovic, A., T. Petrovic, A. Ivanovic, F. Petric, and S. Bogdan. 2021. "A Multi-Resolution Frontier-Based Planner for Autonomous 3D Exploration." *IEEE Robotics and Automation Letters* 6, no. 3: 4528–4535.

Bitcraze. 2019. Crazyflie 2.1. https://www.bitcraze.io/products/old-products/crazyflie-2-1/.

Bitcraze. 2023. Ai-deck 1.1. https://www.bitcraze.io/products/ai-deck/.

Bojarski, M., D. Del Testa, D. Dworakowski, et al. 2016. "End to End Learning for Self-Driving Cars." *arXiv:1604.07316 [cs.CV]*. In press. https://doi.org/10.48550/arXiv.1604.07316.

Chakravarty, P., K. Kelchtermans, T. Roussel, S. Wellens, T. Tuytelaars, and L. V. Eycken. 2017. " CNN-Based Single Image Obstacle Avoidance on a Quadrotor." In IEEE International Conference on Robotics and Automation (ICRA), 6369–6374.

Chang, Y. 2024. *Mulvprl Project*. https://github.com/yingxiu-chang/mulvprl.

Chang, Y., Y. Cheng, U. Manzoor, and J. Murray. 2023a. "A Review of UAV Autonomous Navigation in GPS-Denied Environments." *Robotics and Autonomous Systems* 170: 104533.

Chang, Y., Y. Cheng, J. Murray, S. Huang, and G. Shi. 2022. "The HDIN Dataset: A Real-World Indoor UAV Dataset With Multi-Task Labels for Visual-Based Navigation." *Drones* 6, no. 8: 202.

Chang, Y., Y. Cheng, J. Murray, U. Manzoor, M. Khalid, and F. Cao. 2023b. " Applying and Exploring Supervised Contrastive Regression for Vision-Based Steering Prediction." In International Conference on Automation and Computing (ICAC), 1–6.

Chaplot, D. S., D. Gandhi, S. Gupta, A. Gupta, and R. Salakhutdinov. 2020. "Learning to Explore Using Active Neural SLAM." *arXiv:2004.05155 [cs.CV]*. In press. https://doi.org/10.48550/arXiv.2004.05155.

Chen, S., W. Zhou, A. -S. Yang, H. Chen, B. Li, and C. -Y. Wen. 2022. "An End-to-End UAV Simulation Platform for Visual SLAM and Navigation." *Aerospace* 9, no. 2: 48.

Chen, T., S. Kornblith, M. Norouzi, and G. Hinton. 2020. " A Simple Framework For Contrastive Learning Of Visual Representations." In International Conference on Machine Learning (ICML), 1597–1607.

Chhikara, P., R. Tekchandani, N. Kumar, V. Chamola, and M. Guizani. 2021. "DCNN-GA: A Deep Neural Net Architecture for Navigation of UAV in Indoor Environment." *IEEE Internet of Things Journal* 8, no. 6: 4448–4460.

Doukhi, O., and D. J. Lee. 2022. "Deep Reinforcement Learning for Autonomous Map-Less Navigation of a Flying Robot." *IEEE Access* 10: 82964–82976.

Du, S., H. Guo, and A. Simpson. 2019. "Self-Driving Car Steering Angle Prediction Based on Image Recognition." *arXiv:1912.05440*. In press. https://doi.org/10.48550/arXiv.1912.05440.

Engel, J., T. Schöps, and D. Cremers. 2014. " LSD-SLAM: Large-Scale Direct Monocular SLAM." In European Conference on Computer Vision (ECCV), 834–849, Zurich.

Esrafilian, O., and H. D. Taghirad. 2016. "Autonomous Flight And Obstacle Avoidance of a Quadrotor by Monocular SLAM." In IEEE International Conference on Robotics and Mechatronics (ICROM), 240–245.

Fu, J., Y. Song, Y. Wu, F. Yu, and D. Scaramuzza. 2023. "Learning Deep Sensorimotor Policies for Vision-Based Autonomous Drone Racing." In IEEE International Conference on Intelligent Robots and Systems (IROS), 5243–5250, Detroit.

Güldenring, R., and L. Nalpantidis. 2021. "Self-Supervised Contrastive Learning on Agricultural Images." *Computers and Electronics in Agriculture* 191: 106510.

Güzel, M. S. 2013. "Autonomous Vehicle Navigation Using Vision and Mapless Strategies: A Survey." *Advances in Mechanical Engineering* 5: 234747.

Haresamudram, H., I. Essa, and T. Plötz. 2021. "Contrastive Predictive Coding for Human Activity Recognition." *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5, no. 2: 1–26.

He, K., H. Fan, Y. Wu, S. Xie, and R. Girshick. 2020. " Momentum Contrast for Unsupervised Visual Representation Learning." In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 9729–9738.

He, Y., I. Fang, Y. Li, R. B. Shah, and C. Feng. 2023. "Metric-Free Exploration for Topological Mapping by Task and Motion Imitation in Feature Space." In Robotics: Science and Systems (RSS).

Hornung, A., K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard. 2013. "Octomap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees." *Autonomous Robots* 34: 189–206.

Jain, A., A. Singh, H. S. Koppula, S. Soh, and A. Saxena. 2016. " Recurrent Neural Networks for Driver Activity Anticipation via Sensory-Fusion Architecture." In IEEE International Conference on Robotics and Automation (ICRA), 3118–3125.

Josselyn, S. A., and S. Tonegawa. 2020. "Memory Engrams: Recalling the Past and Imagining the Future." *Science* 367, no. 6473: eaaw4325.

Khairuddin, A. R., M. S. Talib, and H. Haron. 2015. "Review on Simultaneous Localization and Mapping (SLAM)." In IEEE International Conference on Control System, Computing and Engineering (ICCSCE), 85–90.

Khosla, P., P. Teterwak, C. Wang, et al. 2020. "Supervised Contrastive Learning." In Advances in Neural Information Processing Systems (NIPS), 18661–18673.

Kingma, D. P., and J. Ba. 2017. "Adam: A Method for Stochastic Optimization." *arXiv:1412.6980 [cs.LG]*. In press. https://doi.org/10.48550/arXiv.1412.6980.

Koohpayegani, S. A., A. Tejankar, and H. Pirsiavash. 2020. "Compress: Self-Supervised Learning by Compressing Representations." In Advances in Neural Information Processing Systems (NIPS), 12980–12992.

Koppula, H. S., and A. Saxena. 2015. "Anticipating Human Activities Using Object Affordances for Reactive Robotic Response." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38, no. 1: 14–29.

Kouris, A., and C. -S. Bouganis. 2018. "Learning to Fly by Myself: A Self-Supervised CNN-Based Approach for Autonomous Navigation." In IEEE International Conference on Intelligent Robots and Systems (IROS), 1–9.

Laskar, Z., I. Melekhov, S. Kalia, and J. Kannala. 2017. "Camera Relocalization by Computing Pairwise Relative Poses Using Convolutional Neural Network." In IEEE International Conference on Computer Vision Workshops (ICCV), 929–938.

Leyva-Vallina, M., N. Strisciuglio, and N. Petkov. 2023. "Data-Efficient Large Scale Place Recognition With Graded Similarity Supervision." In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 23487–23496.

Leyva-Vallina, M., N. Strisciuglio, and N. Petkov. 2024. "Regressing Transformers for Data-Efficient Visual Place Recognition." In IEEE International Conference on Robotics and Automation (ICRA), 15898–15904.

Li, F., C. Guo, H. Zhang, and B. Luo. 2023. "Context Vector-Based Visual Mapless Navigation in Indoor Using Hierarchical Semantic Information and Meta-Learning." *Complex & Intelligent Systems* 9, no. 2: 2031–2041.

Liang, X., F. Zhu, Y. Zhu, B. Lin, B. Wang, and X. Liang. 2022. "Contrastive Instruction-trajectory Learning for Vision-Language Navigation." In AAAI Conference on Artificial Intelligence, volume 36, 1592–1600.

Loquercio, A., A. I. Maqueda, C. R. Del-Blanco, and D. Scaramuzza. 2018. "Dronet: Learning to Fly by Driving." *IEEE Robotics and Automation Letters* 3, no. 2: 1088–1095.

van der Maaten, L., and G. Hinton. 2008. "Visualizing Data Using T-SNE." *Journal of Machine Learning Research* 9, no. 11: 2579–2605.

McGuire, K., G. De Croon, C. De Wagter, K. Tuyls, and H. Kappen. 2017. "Efficient Optical Flow and Stereo Vision for Velocity Estimation and Obstacle Avoidance on an Autonomous Pocket Drone." *IEEE Robotics and Automation Letters* 2, no. 2: 1070–1076.

McInnes, L., J. Healy, and J. Melville. 2018. "Umap: Uniform Manifold Approximation and Projection for Dimension Reduction." *arXiv:1802.03426 [stat.ML]*. In press. https://doi.org/10.48550/arXiv.1802.03426.

Morin, S., M. Saavedra-Ruiz, and L. Paull. 2023. "One-4-All: Neural Potential Fields for Embodied Navigation." In IEEE International Conference on Intelligent Robots and Systems (IROS), 9375–9382.

Mur-Artal, R., J. M. Montiel, and J. D. Tardós. 2015. "ORB-SLAM: A Versatile and Accurate Monocular Slam System." *IEEE Transactions on Robotics* 31, no. 5: 1147–1163.

Mur-Artal, R., and J. D. Tardós. 2017. "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras." *IEEE Transactions on Robotics* 33, no. 5: 1255–1262.

van den Oord, A., Y. Li, and O. Vinyals. 2019. "Representation Learning With Contrastive Predictive Coding." *arXiv:1807.03748 [cs.LG]*. In press. https://doi.org/10.48550/arXiv.1807.03748.

OpenAI. 2024. Created by Chatgpt. https://chat.openai.com/chat.

Padhy, R. P., S. Verma, S. Ahmad, S. K. Choudhury, and P. K. Sa. 2018. "Deep Neural Network for Autonomous UAV Navigation in Indoor Corridor Environments." *Procedia Computer Science* 133: 643–650.

Palossi, D., F. Conti, and L. Benini. 2019. "An Open Source and Open Hardware Deep Learning-Powered Visual Navigation Engine for Autonomous Nano-UAVs." In IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS), 604–611.

Pan, T., Y. Song, T. Yang, W. Jiang, and W. Liu. 2021. "Videomoco: Contrastive Video Representation Learning With Temporally Adversarial Examples." In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 11205–11214, Nashville.

Park, H. S., J. -J. Hwang, Y. Niu, and J. Shi. 2016. "Egocentric Future Localization." In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 4697–4705.

Parrot. 2012. Ar. drone 2.0. https://www.parrot.com/us/support/documentation/ar-drone.

Ramakrishnan, S. K., Z. Al-Halah, and K. Grauman. 2020. "Occupancy Anticipation for Efficient Exploration and Navigation." In European Conference on Computer Vision (ECCV), 400–418.

Robinson, J., L. Sun, K. Yu, K. Batmanghelich, S. Jegelka, and S. Sra. 2021. "Can Contrastive Learning Avoid Shortcut Solutions?" In Advances in Neural Information Processing Systems (NIPS), 4974–4986.

Ruan, X., and G. Wang. 2021. "Disjoint Contrastive Regression Learning for Multi-Sourced Annotations." *arXiv:2112.15411 [cs.LG]*. In press. https://doi.org/10.48550/arXiv.2112.15411.

Singla, A., S. Padakandla, and S. Bhatnagar. 2019. "Memory-Based Deep Reinforcement Learning for Obstacle Avoidance in Uav With Limited Environment Knowledge." *IEEE Transactions on Intelligent Transportation Systems* 22, no. 1: 107–118.

Song, X., L. Huang, H. Xue, and S. Hu. 2022. "Supervised Prototypical Contrastive Learning for Emotion Recognition in Conversation." *arXiv:2210.08713 [cs.LG]*. In press. https://doi.org/10.48550/arXiv.2210.08713.

Surís, D., R. Liu, and C. Vondrick. 2021. "Learning the Predictability of the Future." In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 12607–12617.

Tang, C. I., I. Perez-Pozuelo, D. Spathis, and C. Mascolo. 2021. "Exploring Contrastive Learning in Human Activity Recognition for Healthcare." *arXiv:2011.11542 [cs.LG]*. In press. https://doi.org/10.48550/arXiv.2011.11542.

Udacity. 2016. *An Open Source Self-driving Car Dataset*. https://www.udacity.com/self-driving-car.

Vemprala, S., S. Mian, and A. Kapoor. 2021. "Representation Learning for Event-Based Visuomotor Policies." In Advances in Neural Information Processing Systems (NIPS), 4712–4724.

Vondrick, C., H. Pirsiavash, and A. Torralba. 2016. "Anticipating Visual Representations From Unlabeled Video." In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 98–106.

Wang, G. 2006. *A Handbook for 1,000 Basic Chinese Characters*. Chinese University Press.

Wang, Y., Y. Jiang, J. Li, et al. 2022. "Contrastive Regression for Domain Adaptation on Gaze Estimation." In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 19376–19385.

Wei, T., P. Lindenberger, J. Matas, and D. Barath. 2024. "Breaking the Frame: Image Retrieval by Visual Overlap Prediction." *arXiv:2406.16204*. In press. https://doi.org/10.48550/arXiv.2406.16204.

Xie, L., A. Markham, and N. Trigoni. 2020. "Snapnav: Learning Mapless Visual Navigation With Sparse Directional Guidance and Visual Reference." In IEEE International Conference on Robotics and Automation (ICRA), 1682–1688.

Xue, Z., and T. Gonsalves. 2023. "Monocular Vision Guided Deep Reinforcement Learning UAV Systems With Representation Learning Perception." *Connection Science* 35, no. 1: 2183828.

Yang, X., J. Chen, Y. Dang, et al. 2021. "Fast Depth Prediction and Obstacle Avoidance on a Monocular Drone Using Probabilistic Convolutional Neural Network." *IEEE Transactions on Intelligent Transportation Systems* 22, no. 1: 156–167.

Yang, X., H. Luo, Y. Wu, Y. Gao, C. Liao, and K. -T. Cheng. 2019. "Reactive Obstacle Avoidance of Monocular Quadrotors With Online Adapted Depth Prediction Network." *Neurocomputing* 325: 142–158.

Youn, W., H. Ko, H. Choi, I. Choi, J. -H. Baek, and H. Myung. 2021. "Collision-Free Autonomous Navigation of a Small UAV Using Low-Cost Sensors in Gps-Denied Environments." *International Journal of Control, Automation and Systems* 19, no. 2: 953–968.

Yu, L., and H. Liu. 2004. "Efficient Feature Selection via Analysis of Relevance and Redundancy." *Journal of Machine Learning Research* 5: 1205–1224.

Yu, X., Y. Rao, W. Zhao, J. Lu, and J. Zhou. 2021. "Group-Aware Contrastive Regression for Action Quality Assessment." In IEEE International Conference on Computer Vision (ICCV), 7919–7928.

Zeng, K. H., W. B. Shen, D. -A. Huang, M. Sun, and J. C. Niebles. 2017. "Visual Forecasting by Imitating Dynamics in Natural Sequences." In IEEE International Conference on Computer Vision (ICCV), 2999–3008.

Zha, K., P. Cao, J. Son, Y. Yang, and D. Katabi. 2024. "Rank-n-Contrast: Learning Continuous Representations for Regression." In Advances in Neural Information Processing Systems (NIPS), 17882–17903.

Zhang, Q., Z. Peng, and B. Zhou. 2022. "Learning to Drive by Watching Youtube Videos: Action-Conditioned Contrastive Policy Pretraining." In European Conference on Computer Vision (ECCV), 111–128.

Zheng, L., Y. Shen, and M. C. Lin. 2022. *Exploring Contrastive Learning With Attention for Self-Driving Generalization*.