DynaNet: Neural Kalman Dynamical Model for Motion Estimation and Prediction

Changhao Chen, *Member, IEEE*, Chris Xiaoxuan Lu, Bing Wang, Niki Trigoni, and Andrew Markham

***Abstract*— Dynamical models estimate and predict the temporal evolution of physical systems. State-space models (SSMs) in particular represent the system dynamics with many desirable properties, such as being able to model uncertainty in both the model and measurements, and optimal (in the Bayesian sense) recursive formulations, e.g., the Kalman filter. However, they require significant domain knowledge to derive the parametric form and considerable hand tuning to correctly set all the parameters. Data-driven techniques, e.g., recurrent neural networks, have emerged as compelling alternatives to SSMs with wide success across a number of challenging tasks, in part due to their impressive capability to extract relevant features from rich inputs. They, however, lack interpretability and robustness to unseen conditions. Thus, data-driven models are hard to be applied in safety-critical applications, such as self-driving vehicles. In this work, we present DynaNet, a hybrid deep learning and time-varying SSM, which can be trained end-to-end. Our neural Kalman dynamical model allows us to exploit the relative merits of both SSM and deep neural networks. We demonstrate its effectiveness in the estimation and prediction on a number of physically challenging tasks, including visual odometry, sensor fusion for visual-inertial navigation, and motion prediction. In addition, we show how DynaNet can indicate failures through investigation of properties, such as the rate of innovation (Kalman gain).**

***Index Terms*— Deep neural network (DNN), dynamical model, motion estimation, state-space model (SSM).**

I. Introduction

FROM catching a ball to tracking the motion of planets across the celestial sphere, the ability to estimate and predict the future trajectory of moving objects is key for interaction with our physical world. With ever increasing automation, e.g., self-driving vehicles and mobile robotics, the ability to not only estimate system states based on sensor data but also to reason about latent dynamics and therefore predict states with partial or even without any observation is of huge importance to the safety and reliability of intelligent systems [1].

Newtonian/classical mechanics has been developed as an explicit mathematical model, which can be used to predict future motion and infer how an object has moved in the past. This is commonly captured in a state-space model (SSM) that describes the temporal relationship and evolution of states through first-order differential equations. For example, in the task of estimating egomotion from visual sensors, also known as visual odometry (VO) [2]–[4], velocity, position, and orientation are usually chosen as physically attributable states for mobile robots. These models are typically handcrafted based on domain knowledge and require significant expertise to develop and tune their hyperparameters. Simplifying assumptions are often made, for example, to treat the system as being linear, time-invariant with uncertainty being additive and Gaussian. A canonical example of an optimal Bayesian filter for linear systems is the Kalman filter [5], which is an optimal linear quadratic estimator. Although capable of controlling sophisticated mechanical systems (e.g., the Apollo 11 lander used a 21-state Kalman Filter [6]), it becomes more challenging to use in complex, nonlinear systems, giving rise to alternative variants, such as the sequential Monte Carlo [7] or nonlinear graph optimization [8]. However, even when using these sophisticated approaches, imperfections in model parameters and measurement errors from sensory data contribute to issues, such as accumulative drift in visual navigation systems. Furthermore, there is a disconnect between the complexities of rich sensor data, e.g., images and derived states.

Identifying the underlying mechanism governing the motion of an object is a hard problem to solve for dynamical systems operating in real world. As a consequence, in recent years, there has been an explosive growth in applying deep neural networks (DNNs) for motion estimation [9]–[17]. These learning-based approaches can extract useful representations from high-dimensional raw data to estimate the key motion states in an end-to-end fashion.

Although these learned models demonstrate good performance in both accuracy and robustness, they are "black-boxes" regressing measurements with implicit latent states and difficult to interpret. In contrast to neural networks, SSMs are able to construct a parametric model description and offer an explicit transition relation that describes the evolution of system states and uncertainty into the future. They can also optimally fuse measurements from multiple sensors based on their innovation gain, rather than simply stacking them as in a neural network.

Manuscript received August 14, 2020; revised July 2, 2021 and September 9, 2021; accepted September 9, 2021. Date of publication September 24, 2021; date of current version December 1, 2021. This work was supported in part by NFSC under Grant 62103427 and Grant 62073331 and in part by the EPSRC Program "ACE-OPS: From Autonomy to Cognitive Assistance in Emergency OPerationS" under Grant EP/S030832/1. *(Corresponding author: Changhao Chen.)*

Changhao Chen is with the College of Intelligence Science and Technology, National University of Defense Technology, Changsha 410073, China (e-mail: chenchanghao@nudt.edu.cn).

Chris Xiaoxuan Lu is with the School of Informatics, University of Edinburgh, Edinburgh EH8 9AB, U.K.

Bing Wang, Niki Trigoni, and Andrew Markham are with the Department of Computer Science, University of Oxford, Oxford OX1 3QD, U.K.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2021.3112460.

Digital Object Identifier 10.1109/TNNLS.2021.3112460

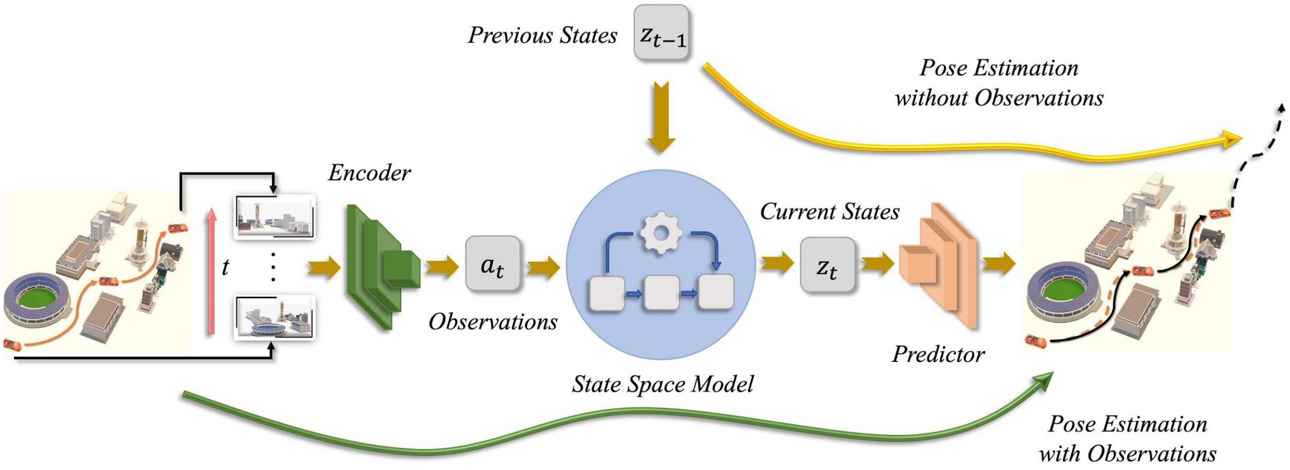


Fig. 1. Concept figure of our proposed DynaNet that combines DNNs and SSMs to infer latent system states. In this case, the motion dynamics of a driving car is modeled by a linear-like dynamical model, with DNNs to extract useful features from visual observations and SSMs to estimate and predict system states with or even without observations.

In this work, we propose DynaNet—neural Kalman dynamical models, combining the respective advantages of DNNs (powerful feature representation) and state-space models (SSMs; explicit modeling of physical processes). As shown in Fig. 1, our proposed end-to-end model enables automatic feature extraction from raw data and constructs a time-varying transition model. The recursive Kalman filter is incorporated to provide optimal filtering on the feature state space and estimate process covariance to reason about system behavior and stability. This allows for accurate system identification in an end-to-end manner without human effort.

Our DynaNet can learn a linear-like SSM directly from raw data. This linear-like structure is a drop-in replacement for the typical recurrent neural network (RNN) estimator. Specifically, our contributions are threefold.

1) We propose a novel hybrid model with differentiable Kalman filter that is adopted on the feature level instead of system states level for latent state inference.
2) We design a strategy to ensure the stability of learned dynamical model by resampling the transition matrix from Dirichlet distribution, in which the system proves to be stable with a probability of one.
3) With the design of neural emission model to connect observations with full states, our DynaNet can cope with a number of challenging situations, e.g., when only partial/corrupted observations are available or even without any observations.

To demonstrate the effectiveness of the proposed technique, we conducted extensive experiments and a systematic study on challenging real-world motion estimation and prediction tasks, including VO, visual-inertial odometry (VIO), and motion prediction. We show how the proposed method outperforms the state-of-the-art deep-learning techniques while yielding valuable interpretable information about model performance. The interpretability analysis discovers the interesting relation between sensor data quality and the explicit model terms.

The rest of this article is organized as follows. Section II reviews the relevant work. Section III presents our proposed neural Kalman dynamical model; Section IV evaluates our proposed model applied to three different tasks, i.e., visual egomotion estimation and prediction, visual-inertial navigation, and motion prediction through extensive experiments. Finally, Section V discusses conclusions.

II. Related Work

A. State-Space Models

SSM is a convenient and compact way to represent and predict system dynamics. In classical control engineering, system identification techniques are widely employed to build parametric SSMs [18], [19]. In order to alleviate the effort of analytic description, data-driven methods, such as Gaussian processes [20] or expectation–maximization (EM) [21], emerged as alternatives to identify nonlinear models. Linear dynamic model, e.g., Kalman filtering, has been explored to combine with RNN that ensures the convergence of neural network training [22]. With advances in DNNs, deep SSMs have been recently studied to handle very complex nonlinearity. Specifically, Backprop KF [23] and DPF [24] used DNNs to extract effective features and feed them to a predefined physical model (i.e., conditioned on algorithmic priors) to improve filtering algorithms. Karkus *et al.* [25] incorporated a particle filter as an algorithmic prior into the neural network for visual localization. Besides feature extraction, DNNs have also been used in reparameterizing the transition matrix in SSMs from raw data [26]–[29]. Unlike prior art, our work exploits recent findings on stable dynamical models [30] and uses resampling to generate a transition matrix from the Dirichlet distribution, whose concentration is learned via a neural network. The specific Dirichlet distribution ensures the stability of dynamic systems, which is an important yet absent property of previous DNN-based SSMs.

B. Motion Estimation

Motion estimation has been studied for decades and plays a central role in robotics and autonomous driving. Conventional VO/SLAM methods rely on multiple-view geometry to estimate motion displacement between images [2]–[4], [31]–[34]. Due to the huge availability and complementary property of inertial and visual sensors, integrating these two sensor modalities has raised increasing attention to give more robust and accurate motion estimates [35]. A large portion of work in this direction is VIO, where filtering [36], [37] and nonlinear optimization [35], [38], [39] are two mainstream model-based methods for sensor fusion. Meanwhile, recent studies also found that the methods using data-driven DNNs are able to provide competitive robustness and accuracy over some model-based methods. These deep learning methods often use convolutional neural networks (ConvNets) to discover useful geometry features from images for effective odometry estimation [11], [16], [17], [40], [41] and/or employ RNNs to model the temporary dependence in motion dynamics [10], [12], [42], [43]. Besides self-motion estimation in robotics and autonomous driving, RNNs have also been introduced to model human dynamics and address the problem of human-skeleton motion prediction [44], [45]. To improve the capacity of long-term prediction, Tang *et al.* [46] leveraged temporal attention mechanism to predict next step motion based on all historical information. Furthermore, Shu *et al.* [47] considered both the spatial and temporal relations of human-skeleton motions and proposed a novel skeleton-joint attention with RNNs to achieve a better performance in the task of human motion prediction. Nevertheless, DNN-based methods are hard to interpret or expect/modulate their behaviors [1]. Motivated by this, our DynaNet aims to bridge the gap of performance and interpretability through a deeply coupled framework of model- and DNN-based methods.

III. NEURAL KALMAN DYNAMIC MODELS

We consider a time-dependent dynamical system, governed by a complex evolving function f

$$\mathbf{z}_t = f(\mathbf{z}_{t-1}, \mathbf{w}_t) \tag{1}$$

where $\mathbf{z} \in \mathbb{R}^d$ is d-dimensional latent state, t is the current timestep, and $\mathbf{w}$ is a random variable capturing system and measurement noise. The evolving function f is assumed to be Markovian, describing the state-dependent relation between latent states $\mathbf{z}_t$ and $\mathbf{z}_{t-1}$. The model in (1) can be reformulated as a linear-like structure, i.e., the state-dependent coefficient (SDC) form, with a time-varying transition matrix $\mathbf{A}$

$$\mathbf{z}_t = \mathbf{A}_t \mathbf{z}_{t-1}. \tag{2}$$

Notably, the system nonlinearity is not restricted by this linear-like structure, as there always exists an SDC form $f(z) = A(z)z$ to express any continuous differentiable function f with $f(0) = 0$ [48].

In this regard, our problem of the dynamic model is how to recover the latent states $\mathbf{z}$ and their time-varying transition relation $\mathbf{A}$ from high-dimensional measurements $\mathbf{x}$ (e.g., a sequence of images), without resorting to a handcrafted physical model.

This work aims to construct and reparameterize this dynamic model using the expressive power of DNNs and explicit state-SSMs. Fig. 2 shows the main framework, which will be discussed in detail in the following. To avoid confusion, in the rest of this article, latent states $\mathbf{z}$ are exclusively used for dynamical models and hidden states $\mathbf{h}$ exclusively represent the neurons in a DNN. In the meantime, we will use sensor measurements and observations interchangeably.

A. Neural Emission Model

Intuitively, the system states containing useful information often lie in a latent space that is different from the original measurements. For example, given a sequence of images (the sensor measurements), the key system states of VO are velocity, orientation, and position. Nevertheless, it is nontrivial for conventional models to formulate a temporal linear model that can precisely describe the relation between these physical representations.

Rather than explicitly specifying physical states as in a classical SSM, we use a DNN to automatically extract the latent state features while forcing them to follow the linear-like relation in (2). This can be achieved automatically by optimizing the model via stochastic gradient descent and backpropagation algorithms. This linearization is particularly useful as it allows us to directly use a Kalman filter for state feature inference. Note that the differentiable Kalman filtering in our DynaNet model performs on the high-dimensional latent feature space rather than the physical state space (e.g., velocity, orientation, and position) as in Backprop KF [23] and DPF [24].

In our neural emission model, an encoder f_{encoder} is used to extract both features $\mathbf{a}_t$ and an estimation of uncertainty $\boldsymbol{\sigma}_t^a$ from the observations $\mathbf{x}_t$ at timestep t

$$\mathbf{a}_t, \boldsymbol{\sigma}_t = f_{\text{encoder}}(\mathbf{x}_t). \tag{3}$$

The features $\mathbf{a}$ act as observations of the latent feature state space. The coupled uncertainties $\boldsymbol{\sigma}$ represent the measurement belief that is transformed into the observation noise $\mathbf{R}$ in a Kalman filter. $\mathbf{a}$ and $\boldsymbol{\sigma}$ are further used in the update stage of a differentiable Kalman filter in (14), which forces them to follow the distribution of KF parameters during end-to-end optimization. Thus, unlike VAE [29], [49], the two terms are not directly and explicitly constrained in (3) to follow a prior distribution but leave the learning model to search for suitable latent states and uncertainty estimation that can construct a linear-like structure and exploit the full capacity of differentiable Kalman filter. However, the observations $\mathbf{a}$ are sometimes unable to provide sufficient information for all latent states $\mathbf{z}$ in a dynamical system, for example, the occasional absences of sensory data. Hence, a deterministic emission matrix $\mathbf{H}$ is defined to connect with the full latent

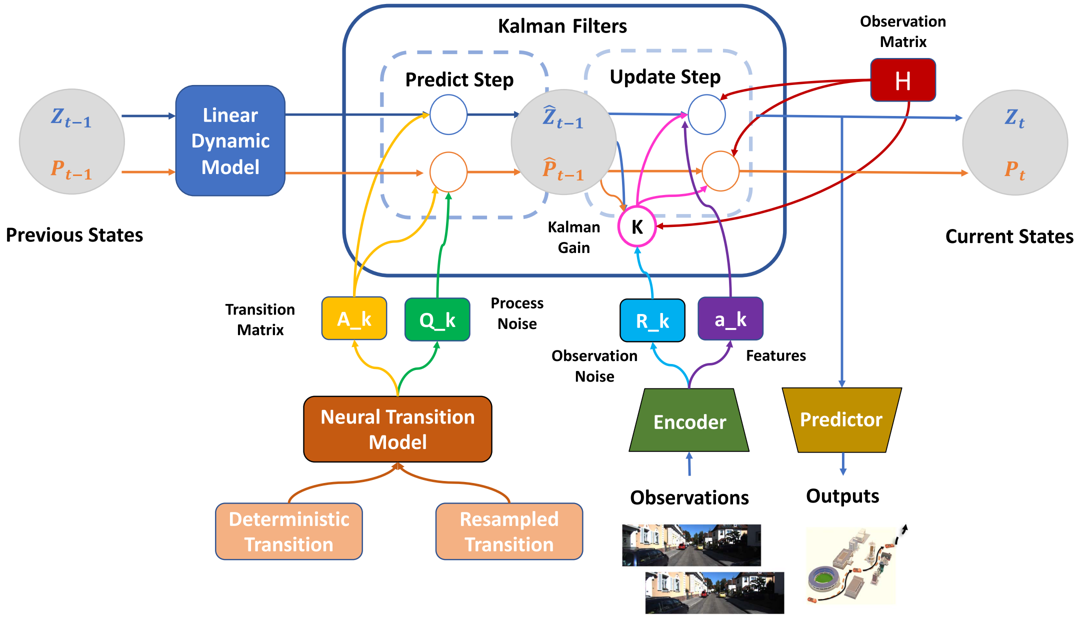


Fig. 2. DynaNet framework consists of the neural observation model to extract latent states **z**, the neural transition model to generate the evolving relation **A**, and a recursive Kalman filter to infer and predict system states. **P** is the covariance matrix of latent states, and **Q** and **R** are process noise matrix and observation noise matrix, respectively.

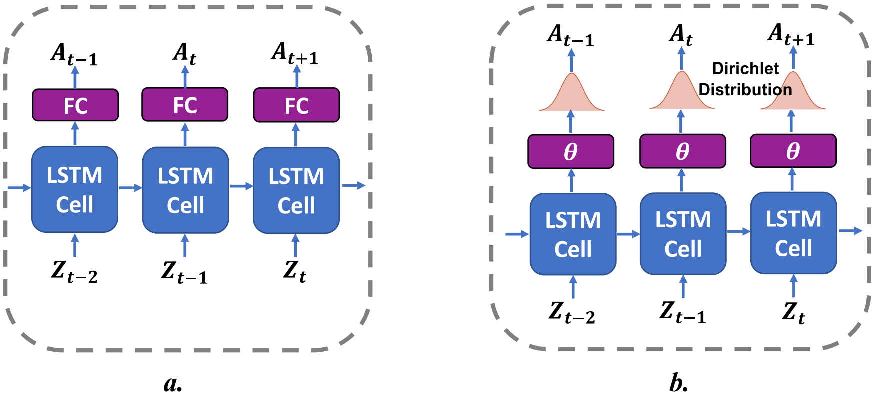


Fig. 3. Two transition generation strategies—(a) deterministic transition or (b) resampled transition are proposed to achieve the desired system behaviors. The transition matrix is generated by an RNN, i.e., LSTM in this work conditioned on the previous latent system states.

states **z**

$$\mathbf{a}_t = \mathbf{H}\mathbf{z}_t. \tag{4}$$

In a practical setting, when the extracted features contain all the information for dynamical systems, the emission matrix **H** is set to d-dimensional identity matrix $\mathbf{I}_d$ as features and states are identical at this moment. On the other side, the identity matrix needs to adapt to $\mathbf{H} = [\mathbf{I}_m, \mathbf{0}_{m\times(d-m)}]$ when observations only give rise to m features. In this case, the rest $(d-m)$ latent states will be attained from historical states. Our experiment in Section IV-C demonstrates the superiority of this neural emission model in addressing the issue of observation absences for sensor fusion in VIO.

B. Neural Transition Model

In an SSM, the temporal evolution of latent states is determined by the transition matrix **A** as in (2). Obviously, the transition matrix is of considerable importance as it directly describes the governing mechanism of a system. Nevertheless, such a matrix is difficult to specify manually, especially when it is time-varying. Fig. 3 shows two methods we propose to estimate **A** on the fly, based on prior system states: 1) a deterministic way to learn it end-to-end from raw data and 2) a stochastic way to resample it from distribution, e.g., the Dirichlet distribution in this work. The deterministic transition and resampled transition are two individual strategies to generate a transition matrix. The parameters of long short-term memory (LSTM) in these two modules are separately learned from data. We will explain them accordingly in what follows.

1) Deterministic Transition: Intuitively, a movement change depends on historical system states, which are encoded in the latent states $\mathbf{z}_{0:t-1}$. Prior works mostly apply an RNN to specify dynamic weights for choosing and interpolating between a fixed number of different transition modes [28], [29]. Inspired by [50], our model generates the transition matrix **A** directly from the history of latent states **z**.

In this deterministic transition model, the dependence of the transition matrix on historical latent states is specified by an RNN. This RNN recursively processes previous hidden states $(\mathbf{z}_{t-1}, \mathbf{h}_{t-1})$ of the dynamic model and LSTM [51], [52] cell and outputs the time-dependent transition matrix $\mathbf{A}_t$ via

$$\mathbf{A}_t = \text{LSTM}(\mathbf{z}_{t-1}, \mathbf{h}_{t-1}) \tag{5}$$

where $\mathbf{z}_{t-1}$ is the latent states of dynamical system at the timestep $(t-1)$, $\mathbf{h}_{t-1}$ is the hidden states of LSTM module, and $\mathbf{A}_t$ is the learned transition matrix at current timestep.

2) Resampled Transition: Powerful DNNs are able to approximate dynamical models effectively from data. However, these learned "black-box" models are difficult to be interpreted or modulated. Especially, in safety-critical applications (e.g., self-driving vehicles), the lack of model behavioral indicator and the absence of system reliability largely limit the adoption of these learning models. The stability of a dynamical system is essential and fundamental to autonomous systems, as it guarantees that the predictions of dynamic models will not change abruptly, given slightly perturbed inputs. Unfortunately, this desirable property is not ensured by most pure DNN models. Therefore, we here aim to ensure the stability of learned dynamical model by resampling the transition matrix from a specific distribution.

The linear-like SSM structure in (2) allows for a quadratic Lyapunov stability analysis, while advances in stochastic optimization allow to construct neural probabilistic models [53]. We thus propose to resample the transition matrix from a predefined probability distribution to enforce the desired stability. Based on the findings in [30], if the state transition follows a Dirichlet distribution in a positive system, it will lead to a model being asymptotically stable, i.e., it will be bounded-input–bounded-output (BIBO) stable. Constructing the stochastic variable and determining the parameters of the distribution are easy to achieve with the widely used reparameterization trick [54].

To further reduce handcrafted engineering, the concentration $\boldsymbol{\alpha}$ of the Dirichlet distribution in our framework is generated from historical system states via an LSTM-based RNN

$$\boldsymbol{\alpha} = \text{LSTM}(\mathbf{z}_{t-1}, \mathbf{h}_{t-1}) \tag{6}$$

where $\mathbf{z}_{t-1}$ is the latent states of dynamical system at the timestep $(t-1)$ and $\mathbf{h}_{t-1}$ is the hidden states of LSTM module. A small Gaussian random noise is also added in this process to improve model robustness. At each timestep, a realization of the transition matrix $\mathbf{A}$ is drawn from the constructed Dirichlet distribution

$$\mathbf{A}_t \sim \text{Dirichlet}(\boldsymbol{\alpha}). \tag{7}$$

Note that in DynaNet, the transition states are in the latent feature space rather than the final target states (e.g., the states of orientation and position in VO). The latent features are extracted by the encoder, which ensures the transition states strictly positive through a rectified linear unit (ReLU) activation and a tiny random positive number on the last layer of the DNN-based encoder.

Our DynaNet extends the work [30] of nearest neighbor method-based stable system to a DNN-based approach, for modeling more complex nonlinear dynamics. Analytically, the following proof supports our proposed system: given a DNN-based dynamical system $\mathbf{z}_{t+1} = \mathbf{A}(\boldsymbol{\alpha})\mathbf{z}_t$, where $\mathbf{z}_t \in \mathbb{R}^d$ is d-dimensional system state at the timestep t, extracted by a neural network from raw data and $\mathbf{A}(\boldsymbol{\alpha}) \in \mathbb{R}^{d\times d}$ is the transition matrix generated by a neural network with a concentration $\boldsymbol{\alpha}$, if the transition matrix $\mathbf{A}$ is constructed from a Dirichlet distribution$\mathbf{A} \sim \text{Dir}(\theta(\boldsymbol{\alpha}))$, this dynamical system is asymptotically stable.

Proof: By resampling the transition matrix $\mathbf{A} \in \mathbb{R}^{d\times d}$ from a Dirichlet distribution, the elements inside $\mathbf{A}$ satisfy

$$\mathbf{A}_{(i,j)} > 0, \quad \mathbf{A}_{(i,j)} < 1 \quad \forall i, j = 1, \ldots, d, \qquad \text{and} \sum_{i=1}^{d}\sum_{j=1}^{d} \mathbf{A}_{(i,j)} = 1. \tag{8}$$

We can have

$$\sum_{j=1}^{d} \mathbf{A}_{(i,j)} < 1 \quad \forall i \implies \max_{i=1:d} \sum_{j=1}^{d} \mathbf{A}_{(i,j)} < 1. \tag{9}$$

If all elements inside $\mathbf{A}$ are strictly positive, then the maximum absolute row sum norm $\|\mathbf{A}\|_\infty$ follows:

$$\|\mathbf{A}\|_\infty = \max_{i=1:d} \sum_{j=1}^{d} |\mathbf{A}_{(i,j)}| < 1. \tag{10}$$

The hidden feature states $\mathbf{z}_{k+M}$ at the timestep $t+M$ is derived from the system states $\mathbf{z}_k$ at the timestep t and M consecutive transition matrix via

$$\|\mathbf{z}_{k+M}\|_\infty = \left\| \prod_{m=1}^{M} \mathbf{A}^m \mathbf{z}_k \right\|_\infty \le \prod_{m=1}^{M} \|\mathbf{A}^m\|_\infty \|\mathbf{z}\|_\infty \le \left(\max_m \|\mathbf{A}^m\|_\infty\right)^M \|\mathbf{z}_k\|_\infty. \tag{11}$$

As the M is infinite, the system states $\mathbf{z}_{k+M}$ will become

$$\|\mathbf{z}_{k+M}\|_\infty \xrightarrow{M\to\infty} 0. \tag{12}$$

Therefore, the system is stable with a probability of one.

C. Prediction and Inference With a Kalman Filter

The neural emission model estimates system states from noisy sensor measurements, while the generated transition model describes the system evolution and predicts the system states with previous ones. However, uncertainties exist in both of them and motivate us to integrate a Kalman filter into our framework. The Kalman filter recursively deals with the uncertainties and produces a weighted average of the state predictions and fresh observations. With the aforementioned neural emission and transition models, the prediction and inference are performed on the feature state space and follow a standard Kalman filtering pipeline. We also note that the Kalman filter's gain controls how much to update the residual error (i.e., the difference between prediction and observation), which is a useful metric to represent the relative quality of measurements (as shown in Section IV-E).

More specifically, the Kalman filter consists of two blocks: prediction and update. In the prediction stage, prior estimates of the mean value and covariance $(\mathbf{z}_{t|t-1}, \mathbf{P}_{t|t-1})$ at the current timestep are derived from the posterior state estimates $(\mathbf{z}_{t-1|t-1}, \mathbf{P}_{t-1|t-1})$ in the previous timestep

$$\begin{aligned}\mathbf{z}_{t|t-1} &= \mathbf{A}_t\mathbf{z}_{t-1|t-1}\\ \mathbf{P}_{t|t-1} &= \mathbf{A}_t\mathbf{P}_{t-1|t-1}\mathbf{A}_t^T + \mathbf{Q}_t.\end{aligned} \tag{13}$$

When current observations $\mathbf{a}_t$ are available, the update process allows us to produce a posterior mean and covariance of hidden states $(\mathbf{z}_{t|t}, \mathbf{P}_{t|t})$ as follows:

$$\begin{aligned}\mathbf{r}_t &= \mathbf{a}_t - \mathbf{H}_t\mathbf{z}_{t|t-1}\\ \mathbf{S}_t &= \mathbf{R}_t + \mathbf{H}_t\mathbf{P}_{t|t-1}\mathbf{H}_t^T\\ \mathbf{K}_t &= \mathbf{P}_{t|t-1}\mathbf{H}_t^T\mathbf{S}_t^{-1}\\ \mathbf{z}_{t|t} &= \mathbf{z}_{t|t-1} + \mathbf{K}_t\mathbf{r}_t\\ \mathbf{P}_{t|t} &= (\mathbf{I} - \mathbf{K}_t\mathbf{H}_t)\mathbf{P}_{t|t-1}\end{aligned} \tag{14}$$

where $\mathbf{r}$ is the residual error (also known as innovation), $\mathbf{S}$ is the residual covariance, and $\mathbf{K}$ is the Kalman gain. In contrast to hand-tuning process noise $\mathbf{Q}$ and measurement noise $\mathbf{R}$ in a conventional KF, these two terms are jointly learned by our proposed neural dynamical model. Finally, the predictor (e.g., an FC network) outputs the target values $\mathbf{y}_t$ from the estimated optimal hidden states $\mathbf{z}_{t|t}$

$$\tilde{\mathbf{y}}_t = f_{\text{predictor}}\big(\mathbf{z}_{t|t}\big). \tag{15}$$

In the case that current measurements, i.e., $\mathbf{a}_t$, are unavailable, the reconstructed values $\hat{\mathbf{y}}_t$ are inferred from the prior estimate $\mathbf{z}_{t|t-1}$

$$\hat{\mathbf{y}}_t = f_{\text{predictor}}\big(\mathbf{z}_{t|t-1}\big). \tag{16}$$

All parameters θ in our model are end-to-end learned with a mean square loss function. This loss function jointly compares the ground truth $\mathbf{y}_t$ with posterior predictions $\tilde{\mathbf{y}}_t$ and prior prediction $\hat{\mathbf{y}}_t$

$$L(\theta) = \frac{1}{T}\sum_{t=1}^{T}\big(||\mathbf{y}_t - \tilde{\mathbf{y}}_t||^2 + ||\mathbf{y}_t - \hat{\mathbf{y}}_t||^2\big). \tag{17}$$

IV. Experiments

We systematically evaluate our system through extensive experiments including pose estimation for VO in Section IV-B, pose estimation for VIO in Section IV-C, and motion prediction without observations or with partial observations in Section IV-D. Moreover, an interpretability study is also conducted in Section IV-E.

A. Datasets, Baselines, and Experiment Setups

1) Datasets: To evaluate our proposed DynaNet models, we used public datasets to conduct experiments: KITTI Odometry dataset [55] for visual pose estimation and prediction, and KITTI Raw Dataset [55] for visual-inertial pose estimation and prediction.

1) *KITTI Odometry Dataset [55]:* It is a commonly used benchmark dataset that contains 11 sequences (00–10) with images collected by car-mounted cameras and ground-truth trajectories provided by GPS. We used it for VO experiment, with Sequences 00–08 for training and Sequences 09 and 10 for testing. The images and ground truth are collected at 10 Hz. We chose the sequence length as 5 and thus generated a total of 20 373 subsequences from the training set to train neural models.
2) KITTI Raw Dataset [55]: It contains both raw images (10 Hz) and high-frequency inertial data (100 Hz). Since inertial data are only available in the unsynced data packages, we selected the raw files with the corresponding KITTI Odometry Dataset. Inertial data and images are manually synchronized according to their timestamps. We adopted the same data split mentioned above, discarding Sequence 03 as its raw data is unavailable. Thus, in this experiment, we used Sequences 00, 01, 02, 04, 05, 06, 07, and 08 for training and Sequences 09 and 10 for testing. We chose the sequence length as 5 and thus generated 20 373 subsequences from the training set to train neural models. We chose the sequence length as 5 and thus generated a total of 20 361 subsequences from the training set to train DNNs.

2) Baselines: In the VO experiment, we compare our DynaNet models with three representative three deep learning-based VO models, i.e., SfmLearner [11], Bian *et al.* [56], and DeepVO [10]. DeepVO shares the same architecture as in our models including the encoder and predictor but uses the two-layer LSTM [52] to estimate system latent states. This can be viewed as an ablation study, and we keep their dimension of hidden states (128) the same as our models for a fair comparison. Except learning-based baselines, our model is also compared with two classical monocular VO system, i.e., ORB-SLAM [34] and VISO2 [57]. ORB-SLAM [34] is a monocular visual SLAM algorithm based on handcrafted features and multiview geometry. Its loop closing module is disabled for a fair comparison with odometry estimation. VISO2 [57] is a monocular VO algorithm and implemented as an official baseline on the KITTI dataset.

In the VIO experiment, we chose a state-of-the-art learning-based VIO approach, i.e., VINet [12] as our baseline. VINet shares a similar structure as in our models, but it uses a two-layer LSTM rather than our DynaNet module. A popular classical model-based VIO system, i.e., mono-VINS [39], is also adopted here as baseline. Mono-VINS [39] is a tightly coupled sliding window-based optimization approach for VIO, which achieves the state-of-the-art performance on several VIO datasets.

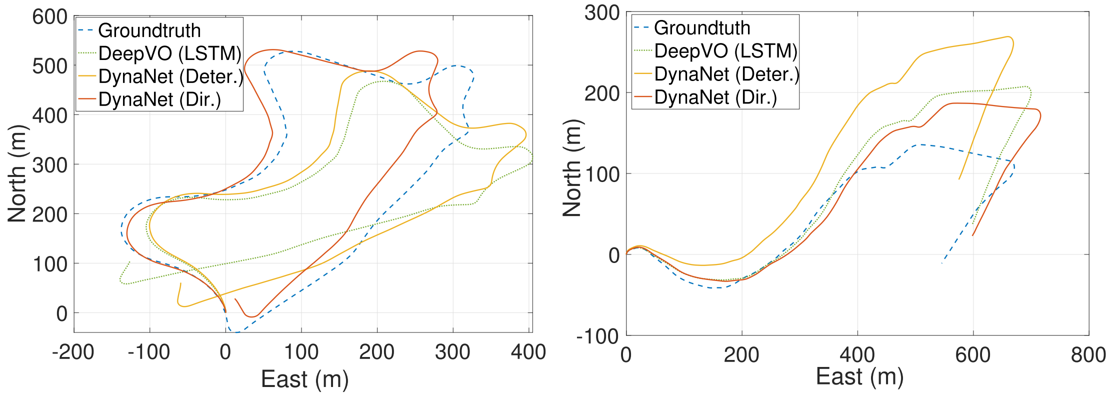
In the motion prediction task, we compare our models with LSTM-based approaches. All the other modules for LSTMs, including encoder and predictor, and the dimension of hidden states (128) are kept the same as in our proposed models for a fair comparison. Besides, we implemented an attention-based LSTM approach to show how attention mechanism improves the motion prediction in future steps. As the approach is an end-to-end model that directly deals with high-dimensional raw images, at each timestep, the attention mechanism aggregates the features extracted from visual data or visual/inertial data and then feeds the updated features to the LSTM module. The rest modules are kept the same to be compared fairly.

TABLE I

PERFORMANCE OF VO ON THE KITTI ODOMETRY DATASET FOR MOTION ESTIMATION, REPORTED IN THE RMSE OF TRANSLATION (%) AND ORIENTATION (°)

	ORB-SLAM	VISO2	SfmLearner	Bian et al.	DeepVO (LSTM)	Ours (Deter.)	Ours (Dirichlet)
09	45.52%, 3.10°	18.06%, 1.25°	17.84%, 6.78°	11.2%, 3.35°	8.01%, 3.10°	6.43%, 2.19°	**4.97%**, **2.10°**
10	**6.39%**, 3.20°	26.10%, 3.26°	37.91%, 15.78°	10.1%, 4.96°	8.53%, 2.41°	8.35%, 2.39°	9.08%, **2.15°**
ave	25.95%, 3.15°	22.08%, 2.25°	27.87%, 11.28°	10.65%, 4.15°	8.27%, 2.75°	7.39%, 2.29°	**7.03%**, **2.12°**

- $t_{rel}(\%)$ is the average translational RMSE drift (%) on lengths of 100m-800m.
- $r_{rel}(°)$ is the average rotational RMSE drift (°/100m) on lengths of 100m-800m.
- The DeepVO (LSTM), our proposed deterministic (Ours (Deter.)) and Dirichlet (Ours (Dirichlet)) model are trained on Sequence 00 - 08 of the KITTI dataset [55] with same hyperparameters for a fair comparison, and tested on Sequence 09 and 10.



(a)

(b)

Fig. 4. Testing trajectories on (a) Sequence 09 and (b) Sequence 10 of the KITTI dataset indicate that our models produce robust and accurate pose estimates in VO.

3) Experiment Setups: We implemented the proposed framework with Pytorch and trained on a NVIDIA Titan X GPU. The implementation details of DynaNet frameworks can be found in the Appendix. All of our models are trained with the Adam optimizer with a batch size of 32 and a learning rate of $1e^{-4}$. The trained models are tested on the test set, in which data have never been seen in the training set.

B. Visual Odometry

Our evaluation starts with a set of visual egomotion (VO) experiments for 6-DoF pose estimation. Here, pose estimation means that our model produces 6-DoF pose given sensor data (i.e., images). In this experiment, a sequence of raw images are given to models to produce pose transformations, i.e., translation and rotation.

Table I reports the performance of our proposed DynaNet models, comparing with other learning-based approaches and classical VO algorithm. All neural networks were trained above the KITTI Odometry dataset with Sequences 00–08 and tested with two new sequences (Sequences 09 and 10). The motion transformations from models are integrated into global trajectories, and then, we evaluated them according to the official KITTI metrics, commonly adopted to evaluate VO algorithms, which calculates the average root-mean-square errors (RMSEs) of the translation and rotation for all the subsequences in the lengths 100, 200, . . . , 800 m. This evaluation metrics can capture both the global and local drifts of VO systems.

As shown in Table I, our proposed models clearly outperform the baselines of ORB-SLAM [34], VISO2 [57], SfmLearner [11], Bian *et al.* [56], and DeepVO [11], and the largest gain is achieved by our Dirichlet model. Note that the only difference between our models and DeepVO is the state estimation part, and we keep the model hyperparameters, e.g., the dimension of hidden states, the same for a fair comparison. By replacing the LSTM module in DeepVO with our proposed DynaNet, our deterministic model improves the performance of DeepVO around 10.64% in translation and 16.73% in orientation, and our Dirichlet model further improves DeepVO around 14.99% in translation and 22.91% in orientation. This indicates that incorporating the physical prior into neural network benefits learning state estimation from data. It also implies that the nonlinearities of VO systems are not lost despite the linear-like structures inside our models.

Fig. 4 shows the trajectories of Sequences 09 and 10 predicted by our models. Sequence 09 and 10 are difficult scenarios, as the driving car experienced large movement in height. Our DynaNet models, especially the Dirichlet model, are still capable of providing robust results, which are closer to the ground-truth trajectories, and consistently show competitive performance over LSTM-based DeepVO.

It must be pointing out that the performance of learning model depends on its training process. Both underfitting and overfitting should be avoided for machine learning methods or their testing performance will be degraded in such circumstances. Fig. 5 shows the trajectories of our proposed DynaNet model with Dirichlet distribution on Sequence 9 of the KITTI dataset in three different training stages. Our model is trained for a total of 100 epochs. When the DynaNet model is trained under the well-fitting condition (Epoch 89),

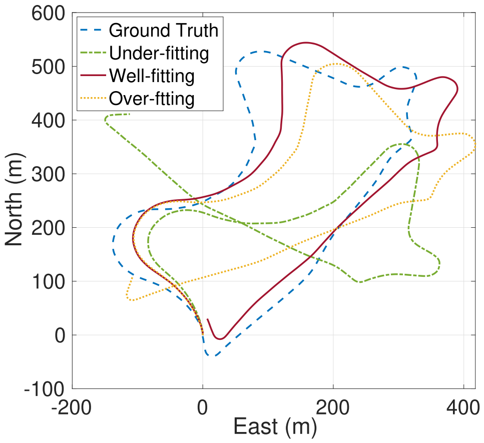


Fig. 5. Testing trajectories of our proposed DynaNet model (Dirichlet) on Sequence 9 of the KITTI dataset in the underfitting, well-fitting, and overfitting conditions.

the testing trajectory is comparable closer to the ground truth. However, when our DynaNet is in the underfitting (Epoch 50) or overfitting (Epoch 90) condition, it sees larger drifts in the corresponding trajectory.

C. Visual-Inertial Odometry

How to effectively integrate and fuse two modalities to provide accurate and robust pose remains a challenging problem. In this experiment, we demonstrate that our proposed models can learn a compact SSM for sensor fusion from two modalities, i.e., visual and inertial data. We also show that our DynaNet models enable robust prediction under the circumstances with partial observations in Section IV-D. When the training model is underfitting or overfitting,

1) Hyperparameters Setup: Initially, a visual encoder and an inertial encoder extract m-dimensional visual features $\mathbf{a}_{\text{visual}} \in \mathbb{R}^m$ and n-dimensional inertial features $\mathbf{a}_{\text{inertial}} \in \mathbb{R}^n$ separately. These two features are then concatenated together as $\mathbf{a} = [\mathbf{a}_{\text{visual}}, \mathbf{a}_{\text{inertial}}] \in \mathbb{R}^{m+n}$. Notably, our emission matrix is defined as identity matrix $\mathbf{H} = \mathbf{I}_{m+n}$ when both modalities are available. If visual or inertial cues are absent, the emission matrix is changed to $\mathbf{H} = [\mathbf{I}_m, \mathbf{0}_{m\times n}]$ or $\mathbf{H} = [\mathbf{0}_{n\times m}, \mathbf{I}_n]$. The training and testing of the visual-inertial dynamic model follows the same procedures as in VO.

2) VIO Pose Estimation: In this experiment, we adopt the official KITTI evaluation metric to evaluate our models and baseline, which is the same as in VO experiments. Table II reports the RMSE of the translation and orientation of the proposed DynaNet models, a classical model-based VIO algorithm, i.e., VINS-Mono [39] and a learning-based approach, i.e., VINet [12]. Due to the problem of loosely time-synchronization between visual and inertial sensors in the KITTI dataset, the performance of VINS-mono is not as good as learning-based methods. This supports the claim that learning-based approaches perform more robustly than hand-designed systems. From Table II, our proposed models outperform VINet with two-layer LSTMs, when given both visual and inertial observations. VINet shares the same framework and hyperparameters as our models, except that it uses LSTM rather than differentiable Kalman filtering. Our deterministic DynaNet [Ours (Deter.)] further reduces the RMSE of VINet from 6.44% to 5.47% in translation and from 1.70° to 1.63°. This demonstrates that our proposed models excel at fusing multiple sensor modalities for more accurate state estimates than the LSTM-based VIO model.

TABLE II

Performance of VIO on the KITTI Raw Dataset for Motion Estimation, Reported in the RMSE of Translation (%) and Orientation (°)

	VINS-Mono	VINet	Ours (Deter.)	Ours (Dir.)
09	41.5%, 2.41°	**3.89%**, 2.02°	5.45%, **1.24°**	4.13%, 1.39°
10	20.3%, 2.73°	8.99%, **1.39°**	**5.49%**, 2.02°	7.03%, 2.64°
ave	30.9%, 2.57°	6.44%, 1.70°	**5.47%**, **1.63°**	5.58%, 2.01°

- $t_{rel}(\%)$ is the average translational RMSE drift (%) on lengths of 100m-800m.
- $r_{rel}(°)$ is the average rotational RMSE drift (°/100m) on lengths of 100m-800m.
- The VINet (LSTM), our proposed deterministic (Ours (Deter.)) and Dirichlet (Ours (Dirichlet)) model are trained on Sequence 00, 01, 02, 04, 05, 07, 08 of the KITTI raw dataset [55] with same hyperparameters for a fair comparison, and tested on Sequence 09 and 10.

TABLE III

VO on the KITTI Odometry Dataset for Motion Prediction [Translation RMSE (0.01 m)]

	5 Steps Prediction	10 Steps Prediction
LSTM (1-layer)	16.8	23.4
LSTM (2-layers)	11.0	17.7
LSTM + Attention	12.1	17.4
Ours (Deterministic)	10.8	16.3
Ours (Dirichlet)	**8.69**	**13.5**

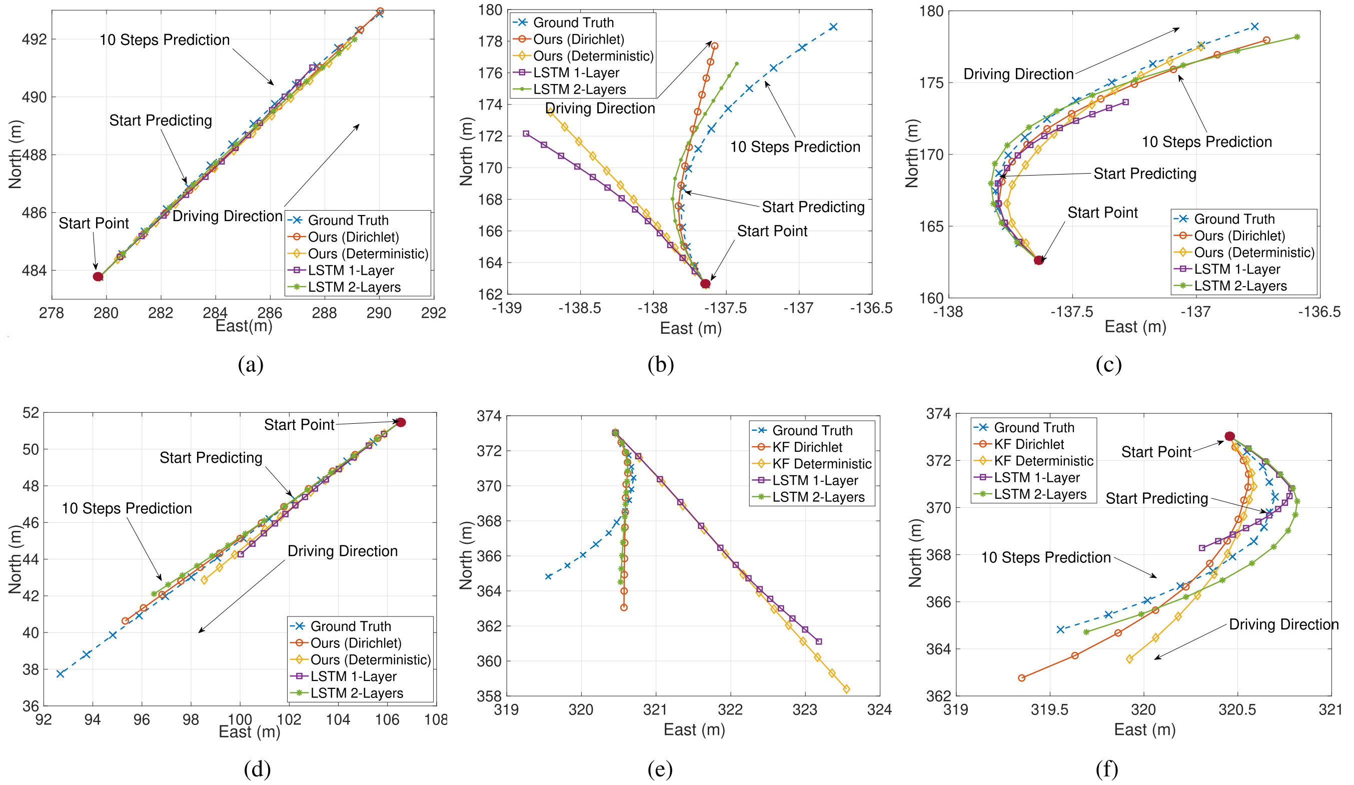
D. Motion Prediction

In this experiment, we show the evaluation of DynaNet models on pose prediction that offers pose without sensor data (i.e., future states prediction).

1) VO Pose Prediction: We first fed VO neural models a sequence of five images for initialization and then let them predict the next five and ten states without any further observations (i.e., trajectory prediction). All models, including baselines, were trained on the training set of the KITTI dataset and tested on the subsequences of 10 or 15 frames, generated from the testing set. In order to compare with LSTM baselines fairly, the structures and hyperparameters of baseline models are kept the same as our models except for the DynaNet module.

Table III shows the quantitative results of our approaches, comparing with LSTM- and "LSTM+Attention"-based models. We report the RMSE of relative positions of the next five and ten steps predicted by neural networks. As shown in Table III, it is clear that our proposed models perform better than both LSTM- and "LSTM+Attention"-based models in visual egomotion prediction. Especially, our Dirichlet model outperforms others by a large margin. This is because the resampled transition matrix from the Dirichlet distribution ensures the learned dynamical model to be stable and hence gives rise to long-term prediction in higher accuracy.

We then demonstrate the qualitative results of our methods. As straight driving routes dominate driving behaviors for autonomous vehicles, we thus begin our performance report with them first. Fig. 6(a) and (d) shows the best case and worst



(a)

(b)

(c)

(d)

(e)

(f)

Fig. 6. For future poses prediction without observations, our Dirichlet-based model clearly outperforms others when predicting the straight driving. Predicted locations in future steps from our proposed Dirichlet-based DynaNet model are closer to the ground truth, compared with other baselines in (a) good case and (d) bad case. In turning, the future poses are estimated in a tangent direction with or without the aid of inertial data in (b) and (c) good case and (e) and (f) bad case.

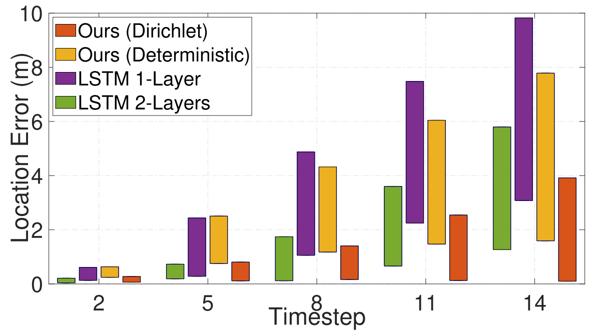


Fig. 7. Error bar of line predictions.

case of line prediction. Clearly, in both cases, the predicted trajectories from our proposed Dirichlet model are much closer to the ground-truth trajectories. Without observations, in the best case, it can even provide the same predictions [the small circles in Fig. 6(a)] as ground-truth values (the small crosses). Fig. 7 shows the error bar of predicted locations for straight driving, with the best and worst results among all tested segments to show the variance of the model predictions. In both cases, our proposed Dirichlet-based model consistently outperforms other baselines by providing more accurate and robust location predictions.

Fig. 6(b) and (e) further shows the prediction performance in turning. As we can see, without timely observation, it is hard for DNNs to estimate accurate orientation changes, but they predict the future poses in a tangent direction in both good and bad cases. In the bad case, the motion predictions are relatively more far away from the ground truth. We will soon show how to integrate inertial information to aid the turning prediction in VIO pose prediction.

2) VIO Pose Prediction: We also evaluate our models on pose prediction using visual and inertial sensor data. This experiment is conducted in scenarios of prediction with visual-only observations, inertial-only observations, and no observations, in which all models are given a sequence of five images for initialization and need to predict the next five poses.

As shown in Table IV, our models, including deterministic and Dirichlet approaches, greatly outperform the comparable LSTM and “LSTM+Attention” approaches. When no input is available, although attention mechanism improves the prediction performance over LSTM, our DynaNet still outperforms this “LSTM+attention” baseline. Specifically, the Dirichlet model shows better performance than all other approaches. This is consistent with the results in visual pose prediction and validates that the model stability will allow more accurate long-term prediction.

Fig. 6(c) and (f) shows the predicted trajectories with only inertial data when no visual observation is given. Clearly, in the good case, our models are capable of predicting future pose evolution accurately and robustly. We note that robust fusion is important for safe operation with missing sensor inputs, e.g., for self-driving cars. In the bad case, though the results are not desirable, the motion predictions still indicate the tendency of turning.

TABLE IV

VISUAL-INERTIAL NAVIGATION ON THE KITTI RAW DATASET FOR MOTION PREDICTION [TRANSLATION RMSE (0.01 m)]

	Prediction w/o Inputs	Visual Only Prediction	Inertial Only Prediction
LSTM (1-layer)	32.5	7.86	23.7
LSTM (2-layers)	21.2	6.90	24.3
LSTM + Attention	17.4	7.54	23.1
Ours (Deterministic)	13.0	**6.27**	**11.3**
Ours (Dirichlet)	**12.3**	6.40	12.3

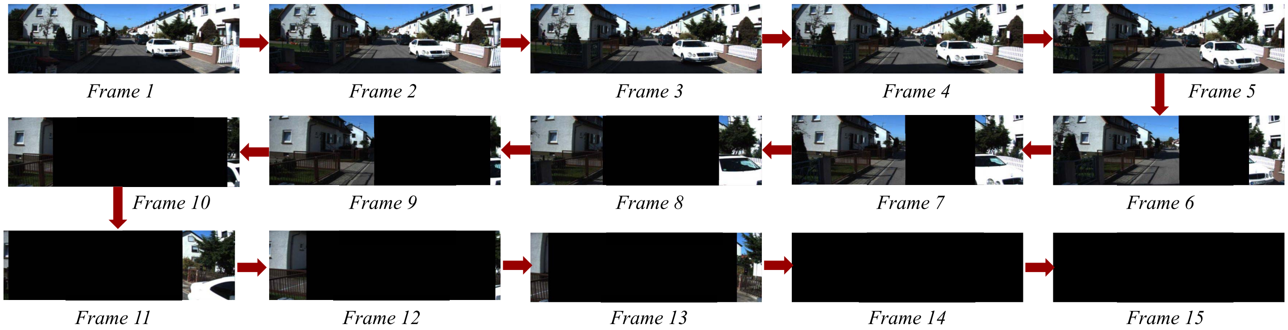
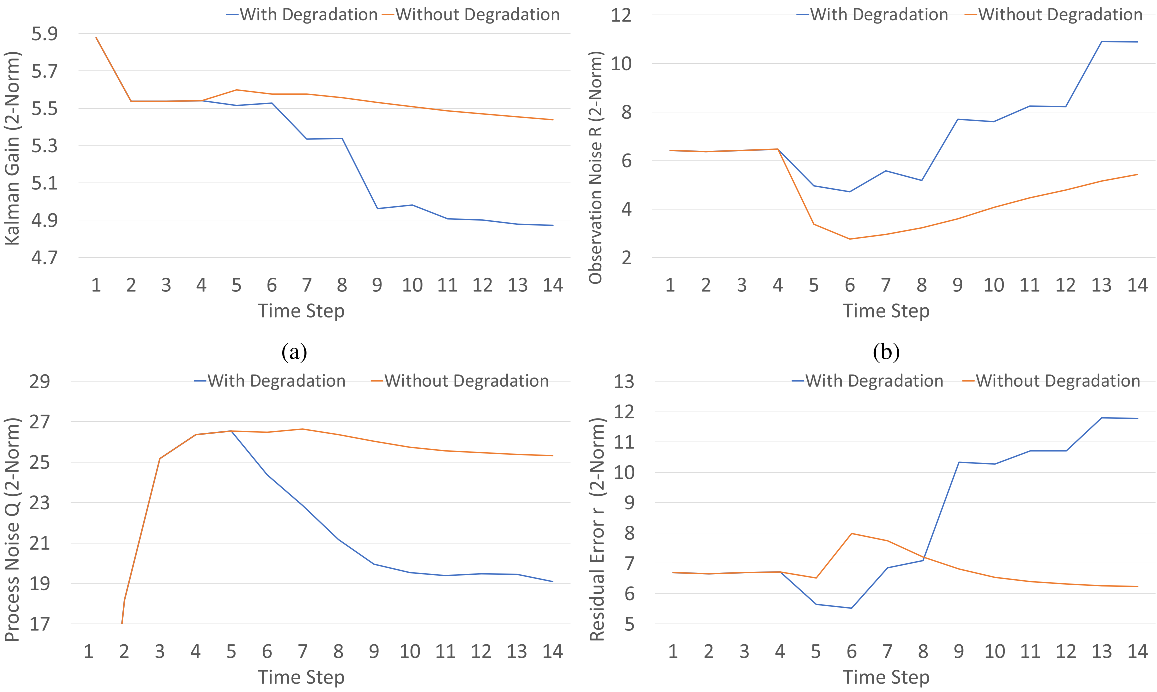


Fig. 8. Sample images from the generated subsequences degraded with increasing size of blanked block in the interpretability experiment.



(a) (b)

(c) (d)

Fig. 9. Model Interpretation. (a) Kalman gain reflects the measurement qualities, which decreases with the rising degree of data corruption. (b) Increasing observation noise and (d) residual noise reflect the rise of observation uncertainty. (c) Decreasing process noise indicates that the model places more trust in process prediction when observations are uncertain.

E. Toward Model Interpretability

We are now in a position to discuss model interpretability. Recall that in the update process (see Section III-C), the Kalman gain is an adaptive weight that balances the observations and the model predictions. If there is a high confidence in measurements, the Kalman gain will increase to selectively upweight measurement innovation and vice versa. This property gives us a unique opportunity to analyze model behaviors from the value changes of the Kalman gain.

To this end, we deliberately fed our Dirichlet model with degraded images and use the Kalman gain to capture the belief in measurements. This experiment generated 113 sequences with 15 frames of images from Sequence 09 in the KITTI dataset. For data degradation, a block was blanked

TABLE V

IMPLEMENTATION DETAILS FOR THE VO EXPERIMENT

Visual Encoder	
[input]	Two stacked images: $B \times 640 \times 192 \times 6$
[layer 1]	Conv. 7^2, Stride 2^2, Padding 3, LeakyReLU activ.
[layer 2]	Conv. 5^2, Stride 2^2, Padding 2, LeakyReLU activ.
[layer 3]	Conv. 5^2, Stride 2^2, Padding 2, LeakyReLU activ.
[layer 3_1]	Conv. 3^2, Stride 1^2, Padding 1
[layer 4]	Conv. 3^2, Stride 2^2, Padding 2, LeakyReLU activ.
[layer 4_1]	Conv. 3^2, Stride 1^2, Padding 1
[layer 5]	Conv. 3^2, Stride 2^2, Padding 2, LeakyReLU activ.
[layer 5_1]	Conv. 3^2, Stride 1^2, Padding 1
[layer 6]	Conv. 3^2, Stride 2^2, Padding 1
[layer 7_1]	FC 128
[layer 7_2]	FC 128, ReLU
[output1]	Observation $\mathbf{a}$: $B \times 128$ (layer 7_1)
[output2]	Observation Noise Covariance $\mathbf{R}$: $B \times 128$ (layer 7_2)
Neural Transition Generation - Deterministic	
[input]	Latent States z: $B \times 1 \times 128$
[layer 1]	LSTM, 1-layer, hidden size 128
[layer 2_1]	FC 128
[layer 2_2]	FC 128, ReLU
[output 1]	Transition Matrix $\mathbf{A}$: $B \times 128$ (layer 2_1)
[output 2]	Process Noise Covariance $\mathbf{Q}$: $B \times 128$ (layer 2_2)
Neural Transition Generation - Resampled	
[input]	Latent States z: $B \times 1 \times 128$
[layer 1]	LSTM, 1-layer, hidden size 128
[layer 2_1]	FC 128, ReLU
[layer 2_2]	FC 128, ReLU
[layer 3]	Dirichlet Distribution
[output 1]	Transition Matrix $\mathbf{A}$: $B \times 128$ (layer3)
[output 2]	Process Noise Covariance $\mathbf{Q}$: $B \times 128$ (layer 2_2)
Kalman Filter Pipeline	
[input1]	Previous Latent States $\mathbf{z}_{t-1}$: $B \times 128$
[input2]	Previous State Covariance $\mathbf{P}_{t-1}$: $B \times 128$
[input3]	Transition $\mathbf{A}$ and Process Noise $\mathbf{Q}$: $B \times 128$, $B \times 128$
[input4]	Observation $\mathbf{a}$ and Observation Noise $\mathbf{R}$: $B \times 128$, $B \times 128$
[layer 1]	Kalman Filter
[output 1]	Current Latent States $\mathbf{z}_t$: $B \times 128$
[output 2]	Current State Covariance $\mathbf{P}_t$: $B \times 128$
Pose Predictor	
[input]	Current Latent States $\mathbf{z}_t$: $B \times 128$
[layer 1_1]	FC 3
[layer 1_2]	FC 3
[layer 4]	(layer 1_1) $\oplus$ (layer 1_2)
[output]	Poses: $B \times 6$

TABLE VI

IMPLEMENTATION DETAILS FOR VIO. $\oplus$ DENOTES A CONCATENATION OPERATION

Visual Encoder	
[input]	Two stacked images: $B \times 640 \times 192 \times 6$
[layer 1]	Conv. 7^2, Stride 2^2, Padding 3, LeakyReLU activ.
[layer 2]	Conv. 5^2, Stride 2^2, Padding 2, LeakyReLU activ.
[layer 3]	Conv. 5^2, Stride 2^2, Padding 2, LeakyReLU activ.
[layer 3_1]	Conv. 3^2, Stride 1^2, Padding 1
[layer 4]	Conv. 3^2, Stride 2^2, Padding 2, LeakyReLU activ.
[layer 4_1]	Conv. 3^2, Stride 1^2, Padding 1
[layer 5]	Conv. 3^2, Stride 2^2, Padding 2, LeakyReLU activ.
[layer 5_1]	Conv. 3^2, Stride 1^2, Padding 1
[layer 6]	Conv. 3^2, Stride 2^2, Padding 1
[layer 7_1]	FC 64
[layer 7_2]	FC 64, ReLU
[output1]	Visual Feature $\mathbf{a}_v$: $B \times 64$ (layer 7_1)
[output2]	Visual Noise Covariance $\mathbf{R}$: $B \times 64$ (layer 7_2)
Inertial Encoder	
[input]	IMU sequence: $B \times 10 \times 6$
[layer 1]	FC 32
[layer 2]	B-LSTM, 2-layers, hidden size 32
[layer 2_1]	FC 64
[layer 2_2]	FC 64, ReLU
[output1]	Inertial Feature $\mathbf{a}_i$: $B \times 64$ (layer 2_1)
[output2]	Inertial Noise Covariance $\mathbf{R}$: $B \times 64$ (layer 2_2)
Sensor Fusion	
[input1]	Visual Feature $\mathbf{a}_v$: $B \times 64$
[input2]	Inertial Feature $\mathbf{a}_i$: $B \times 64$
[layer 1]	(input1 $\oplus$ input2)
[output]	Observation $\mathbf{a}$: $B \times 128$

on a sequence of 15 frames of images. The size of blank blocks gradually increases as time evolves until all pixels on an image are blank. Specifically, as shown in Fig. 8, in each sequence, the images are corrupted with an increasing size of blanked blocks on timesteps 1–5 (no blanked block), 6 and 7 (a blanked block with 192 pixels × 192 pixels), 8 and 9 (a blanked block with 192 pixels × 320 pixels), 10 and 11 (a blanked block with 192 pixels × 480 pixels), 12 and 13 (a blanked block with 192 pixels × 520 pixels), and 14 and 15 (a blanked block with 192 pixels × 640 pixels). The position of the blanked block is randomly selected on each image. We then test our model with this modified dataset in the same fashion as the VO experiment described in Section IV-B.

The Frobenius norm (2-norm) of the Kalman gain matrix is calculated and averaged across all sequences as an aggregated indicator of changes in the Kalman gain matrix. Fig. 9(a) shows that compared to the case with no degradation, this indicator gradually decreases with growing data corruption. It implies that our model can adaptively place more trust on model predictions when observing low-quality data and signal to higher control layers that estimation is becoming more uncertain. Similarly, we further visualize other explicit parameters inside our DynaNet model, i.e., the observation noise matrix $\mathbf{R}$, process noise matrix $\mathbf{Q}$, and residual noise matrix $\mathbf{r}$. As shown in Fig. 9(b)–(d), these parameters are also able to capture and reflect the model uncertainty: with respect to the increasing data corruption, the observation noise rises up as well, indicating that the model is more uncertain about the observations. On the contrary, the process noise goes down, showing that the model has to place more belief in the prediction process when the observations are uncertain. Last but not least, the increasing residual noise also aligns with the uncertainty in observations. It is critical to note that data corrupted in this way have never been seen in the training phase.

V. CONCLUSION

DynaNet, a neural Kalman dynamical model, was introduced in this article to learn temporary linear-like structure on latent states. Through deeply coupled DNNs and SSMs, DynaNet can scale to high-dimensional data as well as model very complex motion dynamics in real world. By using the Kalman filter on feature space, DynaNet is able to reason about latent system states, allowing reliable inference and predictions even with missing observations. Furthermore, the transition matrix in our model is sampled from the Dirichlet distribution learned by an RNN, which ensures system stability in the long run. DynaNet is evaluated on a variety of challenging motion-estimation tasks, including single-modality estimation under data corruption, multiple sensor fusion under data absence, and future motion prediction. Experimental results demonstrate the superiority of our approach in accuracy, robustness, and interpretability.

In the future work, it would be interesting to further explore the application of proposed DynaNet model in motion prediction, e.g., evaluating and visualizing future steps predictions in various environments and measuring the robustness and reliability of a learned stable dynamical model compared with an unstable model.

Appendix
Implementation Details of DynaNet Models

This appendix illustrates the implementation details of the experiments in VO and VIO. All the models are trained with 100 epochs.

A. Visual Odometry

Table V reports the framework for the VO, consisting of visual encoder to extract features $\mathbf{a}$ and observation noise matrix $\mathbf{Q}$, neural transition models (deterministic or resampled) to generate transition matrix $\mathbf{A}$ and process noise matrix $\mathbf{R}$, the Kalman filter pipeline to predict and update system states $\mathbf{z}$, and pose predictor to output six-state poses ([Translation, Euler angles]) from systems states. Specifically, our visual encoder used the encoder structure of the FlowNetS architecture.

B. Visual-Inertial Odometry

As shown in Table VI, the framework for VIO used the same neural transition model, Kalman filter, and pose predictor as in VO, except the encoders and sensor fusion part. Table II shows that the 64-D visual and inertial features are extracted from data by visual and inertial encoders, respectively, and concatenated as a 128-D observation in sensor fusion. The inertial encoder used one-layer bidirectional LSTM to process inertial data.

References

[1] N. Sünderhauf *et al.*, "The limits and potentials of deep learning for robotics," *Int. J. Robot. Res.*, vol. 37, nos. 4–5, pp. 405–420, 2018.

[2] D. Nister, O. Naroditsky, and J. Bergen, "Visual odometry," in *IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 1, Jun. 2004, pp. I-652–I-659.

[3] J. Engel, J. Sturm, and D. Cremers, "Semi-dense visual odometry for a monocular camera," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 1449–1456.

[4] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2014, pp. 15–22.

[5] R. E. Kalman, "A new approach to linear filtering and prediction problems," *J. Basic Eng.*, vol. 82, no. 1, p. 35, 1960.

[6] S. G. Mohinder and P. A. Angus, "Applications of Kalman filtering in aerospace 1960 to the present [historical perspectives]," *IEEE Control Syst. Mag.*, vol. 30, no. 3, pp. 69–78, Jun. 2010.

[7] J. S. Liu and R. Chen, "Sequential Monte Carlo methods for dynamic systems," *J. Amer. Stat. Assoc.*, vol. 93, no. 443, pp. 1032–1044, Aug. 1998.

[8] R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G^{2}o: A general framework for graph optimization," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2011, pp. 3607–3613.

[9] R. Clark, S. Wang, A. Markham, N. Trigoni, and H. Wen, "VidLoc: A deep spatio-temporal model for 6-DoF video-clip relocalization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6856–6864.

[10] S. Wang, R. Clark, H. Wen, and N. Trigoni, "DeepVO: Towards end-to-end visual odometry with deep recurrent convolutional neural networks," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 2043–2050.

[11] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1851–1858.

[12] R. Clark, S. Wang, H. Wen, A. Markham, and N. Trigoni, "VINet: Visual-inertial odometry as a sequence-to-sequence learning problem," in *Association for the Advancement of Artificial Intelligence (AAAI)*. San Francisco, CA, USA: AAAI Press, 2017, pp. 3995–4001.

[13] P. Mirowski *et al.*, "Learning to navigate in cities without a map," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2018, pp. 2419–2430.

[14] M. Bloesch, J. Czarnowski, R. Clark, S. Leutenegger, and A. J. Davison, "CodeSLAM–learning a compact, optimisable representation for dense visual SLAM," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2560–2568.

[15] S. Brahmbhatt, J. Gu, K. Kim, J. Hays, and J. Kautz, "Geometry-aware learning of maps for camera localization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2616–2625.

[16] H. Zhan, R. Garg, C. S. Weerasekera, K. Li, H. Agarwal, and I. Reid, "Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 340–349. [Online]. Available: http://arxiv.org/abs/1803.03893

[17] Z. Yin and J. Shi, "GeoNet: Unsupervised learning of dense depth, optical flow and camera pose," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1983–1992.

[18] D. G. Dudley, "Dynamic system identification experiment design and data analysis," *Proc. IEEE*, vol. 67, no. 7, p. 1087, Jul. 1979.

[19] J. Yu, S. Wang, and L. Li, "Incremental design of simplex basis function model for dynamic system identification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 10, pp. 4758–4768, Oct. 2018.

[20] J. Kocijan, A. Girard, B. Banko, and R. Murray-Smith, "Dynamic systems identification with Gaussian processes," *Math. Comput. Model. Dyn. Syst.*, vol. 11, no. 4, pp. 411–424, 2005.

[21] Z. Ghahramani and S. T. Roweis, "Learning nonlinear dynamical systems using an EM algorithm," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 11, no. 1, 1999, pp. 431–437.

[22] X. Wang and Y. Huang, "Convergence study in extended Kalman filter-based training of recurrent neural networks," *IEEE Trans. Neural Netw.*, vol. 22, no. 4, pp. 588–600, Apr. 2011.

[23] T. Haarnoja, A. Ajay, S. Levine, and P. Abbeel, "Backprop KF: Learning discriminative deterministic state estimators," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 4376–4384. [Online]. Available: http://arxiv.org/abs/1605.07148

[24] R. Jonschkowski, D. Rastogi, and O. Brock, "Differentiable particle filters: End-to-end learning with algorithmic priors," in *Proc. RSS*, 2018, pp. 1–9.

[25] P. Karkus, D. Hsu, and W. S. Lee, "Particle filter networks with application to visual localization," 2018, *arXiv:1805.08975*. [Online]. Available: http://arxiv.org/abs/1805.08975

[26] R. G. Krishnan, U. Shalit, and D. Sontag, "Structured inference networks for nonlinear state space models," in *Proc. Assoc. Advancement Artif. Intell. (AAAI)*, 2017, pp. 1–21.

[27] M. Fraccaro, S. K. Sønderby, U. Paquet, and O. Winther, "Sequential neural models with stochastic layers," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 2207–2215.

[28] M. Fraccaro, S. Kamronn, U. Paquet, and O. Winther, "A disentangled recognition and nonlinear dynamics model for unsupervised learning," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 1–10.

[29] M. Karl, M. Soelch, J. Bayer, and P. van der Smagt, "Deep variational Bayes filters: Unsupervised learning of state space models from raw data," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017, pp. 1–13. [Online]. Available: http://arxiv.org/abs/1605.06432

[30] J. Umlauft and S. Hirche, "Learning stable stochastic nonlinear dynamical systems," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2017, pp. 3502–3510.

[31] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 6, pp. 1052–1067, Jun. 2007.

[32] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "DTAM: Dense tracking and mapping in real-time," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 2320–2327.

[33] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2014, pp. 834–849.

[34] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: A versatile and accurate monocular SLAM system," *IEEE Trans. Robot.*, vol. 31, no. 5, pp. 1147–1163, Oct. 2015.

[35] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation," in *Robotics, Science and Systems*. Rome, Italy: The RSS Foundation, 2015. [Online]. Available: http://www.roboticsproceedings.org/rss11/p06.pdf

[36] M. Li and A. I. Mourikis, "High-precision, consistent EKF-based visual-inertial odometry," *Int. J. Robot. Res.*, vol. 32, no. 6, pp. 690–711, 2013.

[37] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct EKF-based approach," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2015, pp. 298–304.

[38] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual-inertial odometry using nonlinear optimization," *Int. J. Robot. Res.*, vol. 34, no. 3, pp. 314–334, 2015.

[39] T. Qin, P. Li, and S. Shen, "VINS-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 1004–1020, Aug. 2018.

[40] C. Tang and P. Tan, "BA-Net: Dense bundle adjustment networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2019, pp. 1–18.

[41] H. J. Kashyap, C. C. Fowlkes, and J. L. Krichmar, "Sparse representations for object- and ego-motion estimations in dynamic scenes," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 6, pp. 2521–2534, Jun. 2021.

[42] P. Mirowski *et al.*, "Learning to navigate in complex environments," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017, pp. 1–16. [Online]. Available: http://arxiv.org/abs/1611.03673

[43] J. F. Henriques and A. Vedaldi, "MapNet: An allocentric spatial memory for mapping environments," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8476–8484.

[44] K. Fragkiadaki, S. Levine, P. Felsen, and J. Malik, "Recurrent network models for human dynamics," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4346–4354.

[45] J. Martinez, M. J. Black, and J. Romero, "On human motion prediction using recurrent neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2891–2900.

[46] Y. Tang, L. Ma, W. Liu, and W.-S. Zheng, "Long-term human motion prediction by modeling motion context and enhancing motion dynamics," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Jul. 2018, pp. 935–941.

[47] X. Shu, L. Zhang, G.-J. Qi, W. Liu, and J. Tang, "Spatiotemporal co-attention recurrent neural networks for human-skeleton motion prediction," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Jan. 12, 2021, doi: 10.1109/TPAMI.2021.3050918.

[48] T. Çimen, *State-Dependent Riccati Equation (SDRE) Control: A Survey*, vol. 17, no. 1. New York, NY, USA: IFAC, 2008.

[49] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2014, pp. 1–14.

[50] S. S. Rangapuram, M. Seeger, J. Gasthaus, L. Stella, Y. Wang, and T. Januschowski, "Deep state space models for time series forecasting," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2018, pp. 7795–7804.

[51] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[52] K. Greff, R. K. Srivastava, J. Koutnìk, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A search space Odyssey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2222–2232, Oct. 2017.

[53] J. Schulman, N. Heess, T. Weber, and P. Abbeel, "Gradient estimation using stochastic computation graphs," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2015, pp. 1–13.

[54] M. Jankowiak and F. Obermeyer, "Pathwise derivatives beyond the reparameterization trick," 2018, *arXiv:1806.01851*. [Online]. Available: http://arxiv.org/abs/1806.01851

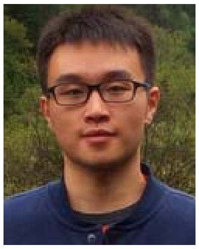
[55] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, 2013.

[56] J. Bian *et al.*, "Unsupervised scale-consistent depth and ego-motion learning from monocular video," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 35–45.

[57] A. Geiger, J. Ziegler, and C. Stiller, "StereoScan: Dense 3D reconstruction in real-time," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2011, pp. 963–968.

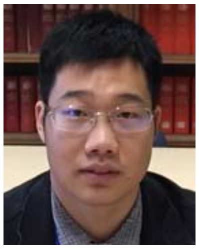
Changhao Chen (Member, IEEE) received the B.Eng. degree from Tongji University, Shanghai, China, in 2014, the M.Eng. degree from the National University of Defense Technology, Changsha, China, in 2016, and the Ph.D. degree from the University of Oxford, Oxford, U.K., in 2020.

He is currently a Lecturer at the College of Intelligence Science, National University of Defense Technology. His research interest lies in robotics, machine learning, and cyberphysical systems.



Chris Xiaoxuan Lu received the M.Eng. degree from Nanyang Technology University, Singapore, in 2015, and the Ph.D. degree from the University of Oxford, Oxford, U.K., in 2019.

He is currently an Assistant Professor at the School of Informatics, University of Edinburgh, Edinburgh, U.K. His research interest lies in cyber-physical systems, which use networked smart devices to sense and interact with the physical world.



Bing Wang received the B.Eng. degree from Shenzhen University, Shenzhen, China, in 2016. He is currently pursuing the Ph.D. degree with the Department of Computer Science, University of Oxford, Oxford, U.K.

His research interest lies in camera localization, feature detection, description and matching, and cross-domain representation learning.

Niki Trigoni is currently a Professor at the Department of Computer Science, University of Oxford, Oxford, U.K. She is also the Director of the EPSRC Centre for Doctoral Training on Autonomous Intelligent Machines and Systems and leads the Cyber Physical Systems Group. Her research interests lie in intelligent and autonomous sensor systems with applications in positioning, healthcare, environmental monitoring, and smart cities.

Andrew Markham received the B.Sc. and Ph.D. degrees from the University of Cape Town, Cape Town, South Africa, in 2004 and 2008, respectively.

He is currently an Associate Professor at the Department of Computer Science, University of Oxford, Oxford, U.K. He is also the Director of the M.Sc. in software engineering. He works on resource-constrained systems, positioning systems, in particular magnetoinductive positioning and machine intelligence.