Image Cognition for Bio-inspired Navigation and Guidance of Autonomous System

By

Wenju Xu

Submitted to the graduate degree program in Department of Aerospace Engineering and the Graduate Faculty of the University of Kansas in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Abstract

How to endorse robots intelligence is consistently a challenging task. The core of robotics is the GNC system containing three dominant tasks: navigation, guidance and control. The traditional methods take redundant measures of different types of sensors to aggregate information on both the robot itself and the environment. Two major issues degrade this solution in recent trend of big data. The first issue is that the information containing in traditional sensors is limited. Bpth IMU and GPS sensor provide low-dimensional data without clue on the environment. The other issue lies in the stability in terms of accessibility and accuracy. Inspired by human and animal vision system, this research instead attempt to improve the GNC system from the visual perception perspective. This is significant in designing autonomous systems to interact within an unknown environment. Different from traditional sensor fusion approaches, this research addresses the robot relevant problems by taking the advantages of computer vision, optimization and deep neural network. This work first proposes a novel optimization based sensor fusion method. It is a direct visual-inertial odometry system that combines visual and inertial measurement for state estimation. To enhance the system with learnable capability, this work then deal with the pose estimation problem in the scheme of deep camera relocalization. It is to train a deep neural network that predicts the camera poses and control commands directly based on the input images. And the down-stream robot perception applications will become possible based on this system. A deep object detection model is thus presented to assist robots in understanding the environment. To further exploit efficiency of deep neural network, we also evaluate different deep learning methods in the end.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Robotics enhanced with the artificial intelligence (AI) is motivated with the concept of autonomy. Nowadays, the academia is working on cutting-edge research. Without the limitation of a special environment for the autonomous robotics, the state of art system is able to work on an open environment. For example, Amazon is delivering packages withing the drone; Google is testing its self-driving cars on the road; Boston dynamic release an amazing video on how its humanoid-like robots walk stably. A few applications are shown in Figure 1.1.

To make the robots working autonomously, there are several challenging problems to be solved. For instance, the drone should first know how its flying condition so that it can make a change accordingly. Furthermore, the autonomous robots should intuitively have the capability to interact with the environment, such as understanding the surroundings. However, no existing system is good enough to take all of these things into consideration. Therefore, it is necessary to equip the autonomous robots with the best technologies at the frontier of sensor fusion, machine learning, and artificial intelligence.

Autonomy is a practical problem, which requires accuracy, reliability and efficiency. The overview of a typical autonomous system is illustrated in Figure 1.2. Navigation, determining the location and orientation as well as velocity at a given time, plays an important role in autonomous system. How to gather accurate information from sensor measurement significantly affects the system efficiency and stability. Kalman filter is the method to fuse IMU and GPS measurement for state estimation. However, These two types of sensors are becoming inadequate for nowadays

Figure 1.1: Application of Autonomous Systems. (top left:) Human is in the new era of autonomous systems; (top right:) A Drone is used to delivery packages by Amazon; (Bottom left:) Different advanced robots introduced by Boston Dynamics; (Bottom right) Self-driving car equipped with sensors.

application. One major issue lies in the fact that the IMU and GPS data do not contain any information of the environment. An another big issue is the intrinsic shortages of the sensors, such as the GPS accessibility and IMU noise.

This work replaces traditional sensors IMU with a new approach to tap into the cognitive behavior of pilots or birds in estimation/understanding attitude information. 1. This work utilizes the camera as the main measurement device. As it contains high-dimensional data, its image data reflect meaningful information on the environment. 2. A optimization based method is proposed to fuse the IMU and camera measurement. IMU measurements can augment visual odometry to remedy this issue by providing a short term measurement, making them an ideal combination to better estimate the pose of the camera and generate information on the surroundings. 3. To enhance the image perception, This work prefer Deep Neural Networks (DNNs) for pose estimation, which is an end-to-end method. It maps the input images to any measure space after the training process.

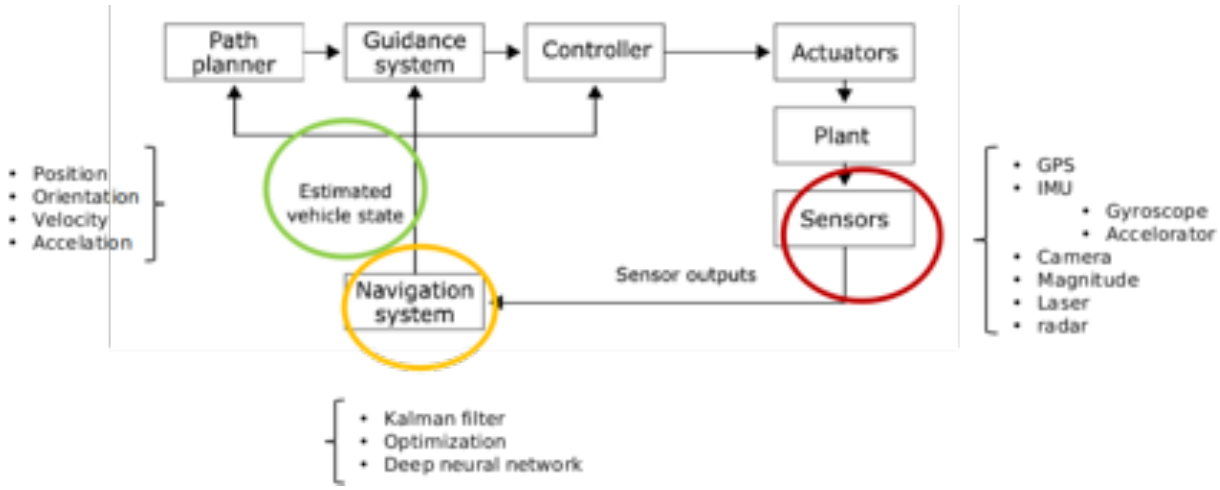


Figure 1.2: Overview of the GNC system.

Specifically, The DNNs enable the aircraft to uses onboard camera. It extracts feature from the the image then estimates roll and pitch angles and their rates, finally to generate appropriate control commands to control the aircraft.

1.2 Sensors

Currently, the state estimation for the autonomous robots is built up on multiple sensor data, such as the IMU, camera, radar, laser scanners, etc, instead of any single one. These redundant measurements guarantee the accuracy and stability of navigation. As a tool for measure, sensors are designed to be sensitive to a certain change, but are also vulnerable to the environment. To the remedy, different types of measurements are fused to reduce the noise. The GPS provides accurate messages, such as location, velocity, accelerate and angular rate at a low frequency (5 Hz). While IMU updates its noisy measurement at a much higher rate (300 Hz). The fusion of these two types of sensors are able to overcome the shortages in each single sensor and leads to high frequency, high accuracy measurements. Different from other sensors, the camera records high dimensional data that intuitively reflects the real world. More importantly, cameras' low price make them an integral part of day by day activities and most of UAS research projects. Thus the images are available in a large scale. How to extract information from a high dimensional data

(a) Camera (b) IMU (c) GPS

Figure 1.3: Illustration of different sensors.

and How cameras must be set up to achieve better performance are not trivial tasks. Another fact is the available computational power of-board that significantly shorten the time in handling large scale data, such as images. So processing the information from the real-time record of an onboard camera is promising.

1.3 Pose Estimation

Past fourthly years, Kalman Filters have been used to estimate attitude information. The fusion scheme follows a prediction and correction process. Different types of measures are connected through a dynamic model. This scheme fuse the data measure in a loosely coupled way. It is efficient when the state vector is low-dimensional.

This thesis presents a direct visual-inertial odometry system. In particular, a tightly coupled nonlinear optimization based method is proposed by integrating the recent development in direct dense visual tracking of camera and the inertial measurement unit (IMU) pre-integration. Then a factor graph optimization is adopted to estimate the pose and position of the camera, and a semi-dense map is created simultaneously. Two sliding windows are maintained in the proposed approach. The first one, based on direct sparse odometry (DSO), is to estimate the depths of candidate points for mapping and dense visual tracking. In the second one, measurements of both the IMU pre-integration and direct dense visual tracking are fused probabilistically based on a

tightly-coupled, optimization-based sensor fusion framework. As a result, the scale drift of visual odometry is compensated by the constraints from the IMU pre-integration. Evaluations on real-world benchmark datasets show that the proposed method achieves competitive results in indoor scenes.

1.4 Deep pose estimation

Deep neural network is a powerful model to infer information. The prospective in cheap camera and powerful GPU promotes to develop deep models for 6 degree pose estimation. Give a sequence images, the deep model succeed in predicting the position and orientation. And other robot perception applications will become possible based on this system. However, the fact is that current models are inadequate in capturing the long-term temporal dependencies appropriately and select the relevant driving series to make predictions.

We present a hybrid deep learning method for camera relocalization. The proposed system leverages the discriminative deep image representation from a convolutional neural networks to capture the long-term temporal dependencies appropriately and select the relevant driving series to make predictions of the six degree of freedom (6DoF) camera pose in an end-to-end fashion. Specifically, we formulate a novel spatial Attention CNN model for joint learning of soft pixel attention and hard graph constraints with simultaneous optimization of feature representations, dedicated to localize the discriminative parts which generates high-quality 6-degree pose estimation. The results show that compared to the state-of-the-art Bayesian camera relocalization method, our model produces comparable localization accuracy.

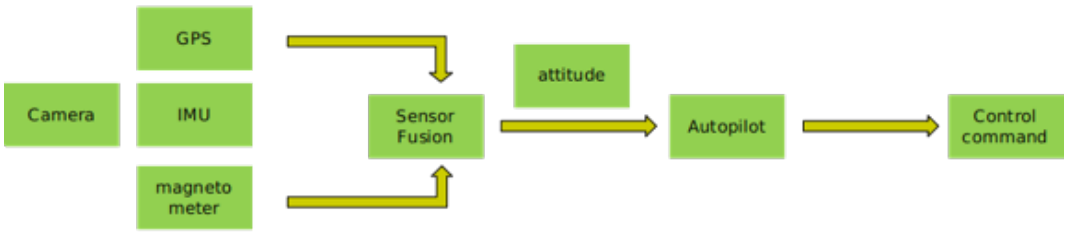


Figure 1.4: Overview of traditional pilot.

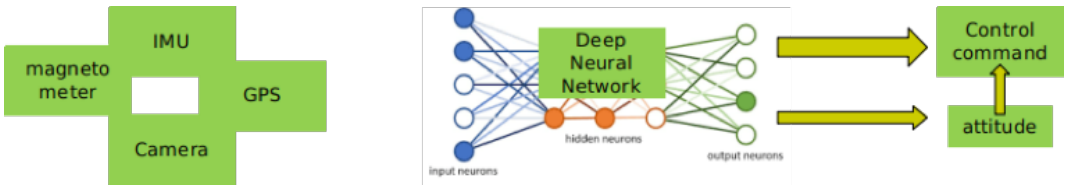


Figure 1.5: Overview of deep nerual network based pilot.

1.5 Deep pilot

Traditionally control command is created based on a well formulated dynamic model. Given the measured state and desired state, the controller is able to generate a sequence of control rule that empowers the system to follow the previously defined movement. However, it is nontrivial to exactly formulate a dynamic model. The recent deep neural network is becoming popular due to its capability of modelling nonlinear system. Without an exactly dynamic model, it is worth to design a deep neural network based controller in an end-to-end manner. Specifically, The deep neural network makes predictions directly on the sensor data, such as images. This model autonomously extracts feature from the inputs then estimates roll and pitch angles and their rates, finally to generate appropriate control commands to control the aircraft. The difference between the traditional pilot and the deep neural network based pilot is demonstrated in Figure 1.4 and 4.7.

Chapter 2

Background and Literature Review

We start by defining the basic notations used in the following sections. These notations will cover different applications but are mainly for sensor fusion and deep learning tasks. Then we review the state-of-the-art models on sensor fusion and the recent trend of utilizing the deep neural network for existing research tasks.

2.1 Notation

The state space in an autonomous system dominated by position p_k^G, velocity v_k^G and attitude θ_k^G, where G stands for the global coordinate system and k indicates the time point. In the sensor fusion task, we take the following notation: $\hat{m}_{k+1}^k$ is the IMU measurements between the k-th and $(k+1)$-th images. $\hat{m}_c^{ref(c)}$ is the dense tracking result for the current image c with respect to the corresponding reference image $ref(c)$ (a key frame). $r_{IMU}(\hat{m}_{k+1}^k, \pi_{k+1}^k)$ is the residual between the IMU integration and state π_{k+1}^k, and $r_I(\hat{m}_c^{ref(c)}, \pi_c^{ref(c)})$ is the dense tracking residual between $\hat{m}_c^{ref(c)}$ and state $\pi_c^{ref(c)}$. Σ_{IMU} and Σ_I are the associated covariances of the IMU measurement and the image alignment. We take the logarithm map: $\theta_k^G = log(R_k^G)^\vee$. Our system maintains several states during processing time. Besides the position, velocity, attitude, the variables to be estimated comprise accelerometer bias b_a and gyroscope bias b_g. The full state space is defined as:

$$\pi_k^G = \begin{bmatrix} p_k^G & v_k^G & \theta_k^G & b_a & b_g \end{bmatrix}$$
$$\pi = \begin{bmatrix} \pi_0^G & \cdots & \pi_k^G & \cdots & \pi_n^G \end{bmatrix}$$

where π_k^G is the k-th state written in the global coordinate system; n is the number of the frames in the optimization set.

In the deep learning task, we take $\{X, Y\}$ as a set containing the training images x and corresponding labels y. The model $f(x;\omega)$ parameterized by ω takes in the images data. The training process is find the optimal $\omega^\star$ to minimize the loss function $\ell(f(x;\omega), y)$. We will follow these notations and make a new definition if necessary.

2.2 IMU Model

IMU sensor consists of a accelerator and gyroscope. It is to measure the acceleration and angular acceleration. By integrating these measurements within a period of time, it is able to get the position, velocity and attitude. However, this sensor suffers dramatically the system noise. It is good at short time measurement while the accumulated error degrades the accuracy in a long term. To quantitatively take into account of this err, we need to model the noise in IMU sensor. The measured inertial quantities, including linear acceleration and angular velocity $\begin{bmatrix} \hat{a}_k^G \\ \hat{w}_k^G \end{bmatrix}$, are modeled as sum of the true value $\begin{bmatrix} a_k^G \\ w_k^G \end{bmatrix}$, time-variant bias $\begin{bmatrix} b_a^i \\ b_g^i \end{bmatrix}$, and Gaussian white noise $\begin{bmatrix} \eta_a^k \\ \eta_g^k \end{bmatrix} dt$, which is:

$$\begin{bmatrix} a_k^G \\ w_k^G \end{bmatrix} = \begin{bmatrix} \hat{a}_k^G \\ \hat{w}_k^G \end{bmatrix} + \begin{bmatrix} b_a^i \\ b_g^i \end{bmatrix} + \begin{bmatrix} \eta_a^k \\ \eta_g^k \end{bmatrix} dt$$

The Gaussian white noise can be characterized as: their mean is zero, and their covariances matrices are obtained by integrating the covariances of a_n, w_n, a_w and w_w over the step time dt, which results:

$$V_k = \sigma_{\hat{a}_n}^2 dt^2 I$$

$$\Theta_k = \sigma_{\hat{w}_n}^2 dt^2 I$$

$$A_k = \sigma_{a_w}^2 dt^2 I$$

$$\Omega_k = \sigma_{w_w}^2 dt^2 I$$

Here, V_k , Θ_k,A_k and Ω_k are the random impulses applied to the velocity, orientation and bias estimates, modeled by white Gaussian processes. The constant $\sigma_{\hat{a}_n}$, $\sigma_{\hat{w}_n}$, σ_{a_w} and σ_{w_w} are determined from the IMU datasheet, or from experimental measurements.

2.3 Camera Model and image geometry

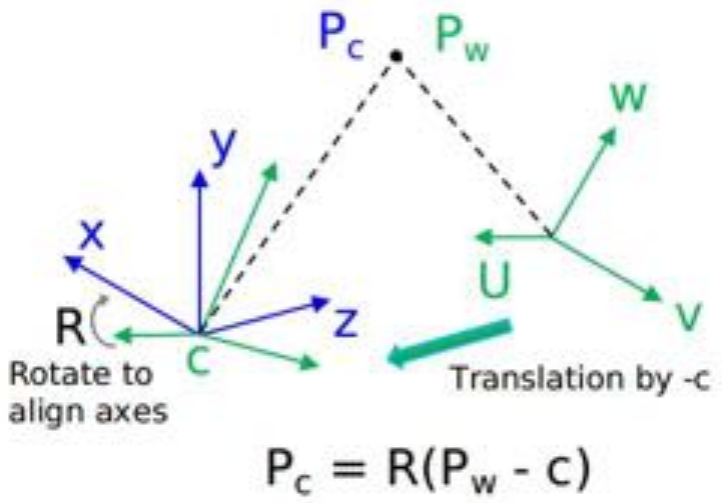


Figure 2.1: Illustration of image geometry.

Images are usually took by a camera with different poses. We takes translation t and rotation R to indicate the motion between two camera poses. This motion is represented by a transformation matrix $T = [R \mid t]$. Figure 2.1 demonstrates the same point captured by a camera in two different

pose. Camera model describes the procedure where a 3D point $X = \begin{bmatrix} x & y & z \end{bmatrix}^T$ is projected to image at $\begin{bmatrix} u & v \end{bmatrix}^T$, The intrinsic parameters of a camera used in our model consist of the focal length f_x, f_y and the camera center c_x , c_y. We project 3D points from the camera coordinate system to the image plane by

$$\begin{bmatrix} u \\ v \end{bmatrix} = w(X) = (\frac{xf_x}{z} + c_x, \frac{yf_y}{z} + c_y)^T$$

where $w(X)$ is the projection function. As for its inverse, we use

$$X = w^{-1}(u,d) = (\frac{u - c_x}{f_x}d, \frac{v - c_y}{f_y}d, d)^T$$

Given a transformation $T = \begin{bmatrix} R^c_{ref(c)} & p^c_{ref(c)} \end{bmatrix}$ from the frame c to its reference frame ref(c) , a pixel $u^{ref(c)}_{ij}$ of the refence frame with the depth d_u can be reprojected to the pixel u^c of frame c by the warp function

$$u^c = w(R^c_{ref(c)} w^{-1}(u^{ref(c)}_{ij}, d_u) + p^c_{ref(c)})$$

Camera pose estimation is to estimate the transformation matrix $T = [R \mid t]$ in a global coordinate system. By tracking the relative camera motion between two consecutive images, it is not hard to recover the camera pose in the global coordinate system.

2.4 State Estimation Methods

State estimation is the key part in a navigation system. How to infer from the sensor's noisy measurements the accurate system state and to combine redundant measurements for stable and efficient sensor fusion is challenging. Figure 2.3 list different state estimation methods. The

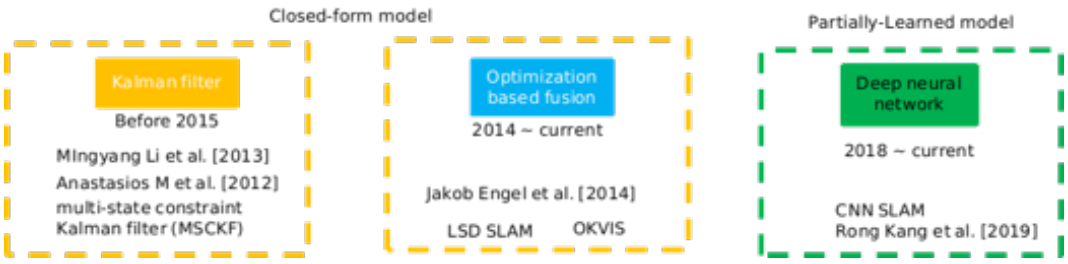


Figure 2.2: Illustration of different state estimation methods.

Kalman filter is the mostly employed method for sensor fusion. Typically, this method employing a prediction-correction scheme is effective in low-dimensional sensor fusion. In dealing with the recent trade of big data, especially, the image and voice data, The kalman filter is not efficient or effective. Recent methods are utilizing the optimization method to find the optimal solution. The previous prediction-correction scheme is reformulated into a one unified equation. The solution is obtained by employing the popular Gauss-Newton or Levenberg-Marquardt algorithms. More recently, deep neural network demonstrates its sophisticated capability to approximate any nonlinear function. The deep neural network is driving by the stochastic gradient decent (sgd) method that iteratively approximates to the optimal solution. It is potential for the deep neural network to learn a mapping between two domains. After a proper training, the deep neural network is able to map the input data to a measure space.

Simultaneously localization and mapping (SLAM) has a long history in monocular scenarios, prominent examples include dense tracking and mapping (DTAM) (100), semi-direct visual odometry (SVO) (39) and large-scale direct (LSD) monocular SLAM (30), which work on sparse features and estimate the camera motion through a prediction and correction fashion. Direct methods work directly on image intensities and attract a lot of attention since they are robust and computational efficient without feature detection. It is demonstrated that direct methods are more suitable for dense mapping than feature based method, and when enhanced by edge alignment (70), they can deal with changing illumination and fast motion. More recently, direct sparse odometry (DSO) (28) presented an impressive semi-dense 3D reconstruction of the environment through a sliding window optimization. This direct method minimizes the photometric error of points with suffi-

ciently high image gradient magnitude (edges) using a non-linear optimization framework.

More recently, people pay great attention to the 3D SLAM, like the semantic SLAM (172) and the deep learning enhanced SLAM (129). Semantic SLAM focuses on simultaneous 3D reconstruction and material recognition and segmentation, which ends up with a real-time end-to-end system. Deep learning is very promising technique in computer vision. (129) takes the advantage of depth prediction from convolutional neural networks (CNNs) to enhance the performance of monocular SLAM. DSO (28) utilizes one monocular camera for 3D reconstruction within an inverse depth estimation framework. It is based on photometric error optimization of windowed sparse bundle adjustment. The following sections will describe different state estimation methods in details.

2.5 Kalman Filter

Given a well formulated dynamic model, Kalman filter is a commonly used sensor fusion method. In the scenarios of state estimation, It assumes the state is in a latent space of the observed sensor data. This method attempts to recover the latent state based on the observation. Given a guess of the initial state, it first makes prediction of the next step state on a dynamic model and correct the prediction with observations from a redundant sensor measurement. Assume that we want to know the value of a variable within a process of the form:

$$x_{k+1} = \Phi x_k + \omega_k$$

where x_k is the current state vector at time k; Φ is the state transition matrix that projects the state at k to new state at k+1. This transition matrix describes the characteristics of the system and is assumed to be unchanged over time; ω is the associated white noise process with known covariance. Observations on this variable can be modelled in the form:

$$z_k = Hx_k + v_k$$

where z_k is the actual sensor measurement of x at time k, H is the noiseless connection between the state vector and the measurement vector, and is assumed unchanged over time; v_k is the associated measurement error.

We assume the covariances of the two noise models are the internal characteristics of the model and sensor, and they are keeping the same values over time. They are statistically formulated as:

$$Q = E[\omega_k \omega_k^T]$$

$$R = E[v_k v_k^T]$$

Once we have the predicted state and observation, we correct the state by minimizing the difference between them. The mean squared error is given by:

$$P_k = E[e_k e_k^T] = E[(x - \hat{x}_k)(x_k - \hat{x}_k)^T]$$

Assuming the prior estimate of $\hat{x}_k$ is called $\hat{x}'_k$, and was gained by knowledge of the system. The update equation for the new estimate, combining the old estimate with measurement data is:

$$\hat{x}_k = \hat{x}'_k + K_k(z_k - H\hat{x}'_k)$$

where K_k is the Kalman gain. The term $z_k - H\hat{x}'_k$ is known as the measurement residual that represents the distance between prediction and measurement:

$$i_k = z_k - H\hat{x}'_k$$

Then we can take this residual to update the predictions. The corrected state is given by:

$$\hat{x}_k = \hat{x}'_k + K_k(Hx_k + v_k - H\hat{x}'_k)$$

To find the optimal solution K_k that minimizes the mean squared error, we reform the equation with that the $x_k - \hat{x}'_k$ is the error of the prior estimate and is uncorrelated with the measurement noise

$$\begin{aligned}
P_k &= E[[(I - K_k H)(x_k - \hat{x}'_k) - K_k v_k][(I - K_k H)(x_k - \hat{x}'_k) - K_k v_k]^T] \\
&= (I - K_k H)E[(x_k - \hat{x}'_k)(x_k - \hat{x}'_k)^T](I - K_k H)^T + K_k E[v_k v_k^T]K_k^T \\
&= (I - K_k H)P'_k(I - K_k H)^T + K_k R K_k^T \\
&= P'_k - K_k H P'_k - P'_k H^T K_k^T + K_k(H P'_k H^T + R)K_k^T
\end{aligned}$$

The potential solution of K_k is from the saddle points. we first take the differentiation with respect to K_k, which leads to:

$$\frac{dP_k}{dK_k} = -2(HP'_k)^T + 2K_k(HP'_k H^T + R)$$

Put this to be zero and reforming it leads to:

$$(HP'_k)^T = K_k(HP'_k H^T + R)$$

$$K_k = P'_k H^T (HP'_k H^T + R)^{-1}$$

We then take it as the solution of K_k and put it back to equation of the mean squared error. We can update the error based on the observation at time k:

$$\begin{aligned}
P_k &= P'_k - P'_k H^T (HP'_k H^T + R)^{-1} HP'_k \\
&= P'_k - K_k H P'_k \\
&= (I - K_k H)P'_k
\end{aligned}$$

Based on the transition matrix Φ, we project the current state to the prior state at the next time

as:

$$\hat{x}'_{k+1} = \Phi \hat{x}_k$$

To complete the recursion it is necessary to find an equation which projects the error covariance matrix into the next time interval:

$$\begin{aligned} e'_{k+1} &= x_{k+1} - x'_{k+1} \\ &= (\Phi x_k + \omega_k) - \Phi \hat{x}_k \\ &= \Phi e_k + \omega_k \end{aligned}$$

Reforming the equation with that the e_k is uncorrelated with the noise ω_k

$$\begin{aligned} P'_{k+1} &= E[e'_{k+1} e'^{T}_{k+1}] \\ &= E[(\Phi e_k)(\Phi e_k)^T] + E[\omega_k \omega_k^T] \\ &= \Phi P_k \Phi^T + Q \end{aligned}$$

In summary, the Kalman filter works in a recursion manner. The steps in one circle is given in Table 5.1.

Description	Equation
Kalman Gain	$K_k = P'_k H^T (H P'_k H^T + R)^{-1}$
Correct Estimate	$\hat{x}_k = \hat{x}'_k + K_k(z_k - H\hat{x}'_k)$
Update Covariance	$P_k = (I - K_k H) P'_k$
Predict state at k+1	$\hat{x}'_{k_1} = \hat{\Phi(x)}_k \quad P_{k_1} = \Phi P_k \Phi^T + Q$

Table 2.1: Kalman Filter updating and corrction steps.

2.5.1 Kalman Filter Related Works for State Estimation

GPS, IMU and magnetometer are widely used sensors in a pilot system. Given the dynamic model of an autonomous system, It is not difficult to determine the transition matrix and observation matrix. The kalman filter is an effective and efficient sensor fusion method, especially in the robot research society(15; 56). Recently people combine the machine/deep learning (174) and kalman filter for state estimation (27; 1).

2.6 Optimization Methods

Optimization is another way to predict the state given the observations. This method is different from the kalman filter method in the way that the state is estimated by directly minimizing the distance between the estimation and the measurement. The observation is a function of the latent state.

2.6.1 Optimization Based Estimation

Let $x = (x_1, ..., x_T)^T$ be a sequence of nodes, where x_i describes the state at time i. Let $z_i j$ and Ω_{ij} be the mean and information matrix of the measurement between node i and j. $z_{ij}(\hat{x_i}, x_j)$ represent the prediction. The log-likelihood l_{ij} is

$$l_{ij} \propto [z_{ij} - \hat{z_{ij}}(x_i, x_j)]^T \Omega_{ij} [z_{ij} - \hat{z_{ij}}(x_i, x_j)]$$

the error function is

$$e_{ij}(x_i, x_j) = z_{ij} - \hat{z}_{ij}(x_i, x_j)$$

The optimal state is obtained by minimizing the objective function:

$$x^\star = \underset{x}{\operatorname{argmin}} F(x)$$

where

$$F(x) = \sum_{<i,j>\in C} e_{i,j}^T \Omega_{i,j}^T e_{i,j}$$

The numerical solution can be obtained by using the popular Gauss-Newton or Levenberg-Marquardt algorithms, which takes the taylor expansion of the error function around the current initial guess state $\hat{x}$

$$\begin{aligned} e_{ij}(\hat{x}_i + \Delta x_i, \hat{x}_j + \Delta x_j) &= e_{ij}(\hat{x} + \Delta x) \\ &= e_{ij} + J_{ij}\Delta x \end{aligned}$$

where J_{ij} stands for the Jacobian of $e_{ij}(x)$. We get the solution of F_{ij}

$$\begin{aligned} F_{ij}(\hat{x} + \Delta x) &= e_{ij}(\hat{x} + \Delta x)^T \Omega_{ij} e_{ij}(\hat{x} + \Delta x) \\ &= (e_{ij} + J_{ij}\Delta x)^T \Omega_{ij}(e_{ij} + J_{ij}\Delta x) \\ &= e_i j^T \Omega_{ij} e_{ij} + 2e_{ij}^T \Omega_{ij} J_{ij}\Delta x + \Delta x^t J_{ij}^T \Omega_{ij} J_{ij}\Delta x \\ &= c_{ij} + 2b_{ij}\Delta x + \Delta x^T H_{ij}\Delta x \end{aligned}$$

Taking the second derivative and set to zero. The minimal solution can be obtained as

$$H\Delta x^\star = -b$$

The linearized solution is obtained by:

$$x^\star = \hat{x} + \Delta x^\star$$

2.6.2 optimization based research and applications

Previous research on camera-based robot navigation includes RGBD visual odometry using stereo camera (31) or Kinect (65), and visual inertial odomerty (81; 92; 93; 110). There also are efforts to use monocular visual odometry, often for unmanned aerial vehicle that has strict weight constraints. However, monocular cameras are prone to long-term scale drift. To address this problem, researchers proposed filter-based fusion methods (81; 93) and, more recently, optimization-based methods (86) that combine IMU and visual sensors for better performance.

In these methods, features such as corners (32) and other special patterns are used for image alignment. However, the feature detection and matching largely depends on the quality of images, which can make it unstable to changes in the environment. Recently, researchers introduced visual odometry methods that use direct image alignment (65; 87; 101), that are free of feature detection and matching, and are stable to the environmental change.

Besides point features, edge features are natural and informative which have been receiving great attention (148; 128). It is acknowledged that edge features are more robust to light variance, motion blur and occlusion than point features. Specifically, edges are more prominent than any other information in texture-less scenes. Previous researches mainly use line segments rather than comprehensively utilizing edges for the difficulties with describing and matching edges. There are researches focusing on fusing these methods and compensating for each other. Lu and Song (90) fuse point and line features in RGB-D VO to deal with lighting variations and uneven feature distributions. Forster et al. (39) design a semi-direct method that optimize the photometric error of small patches around FAST corners.

Historically, there have been two main concepts towards approaching the visual-inertial estimation problem: batch nonlinear optimization methods and recursive filtering methods. While the former method jointly minimizes the error originating from integrated IMU measurements and the reprojection errors from visual terms (78), recursive algorithms commonly use the IMU measurements for state propagation while updates originate from the visual observations (118; 81). At the back-end, fusion methods can mainly be divided into two classes. In loosely-coupled fusion

(86; 78), visual measurements are first processed independently to obtain high-level pose information and then fused with inertial measurements, usually using a filtering framework (81; 80). Effectively, two sub-problems are solved separately in loosely-coupled fusion, resulting in a lower computational cost, but results are suboptimal. In tightly-coupled fusion (21; 133), both visual and inertial measurements are fused and optimized in a single framework. It considers the coupling between two types of measurement and allows the adoption of a graph optimization-based framework with iterative re-linearization to achieve better performance. thus, tightly-coupled methods usually come with a higher computational cost.

In our proposed algorithm, we devise a semi-tightly coupled function that combines visual and inertial terms in a fully probabilistic manner. We adopt the concept of keyframes due to its successful application in classical vision-only approaches: it is implemented using partial linearization and marginalization. The keyframe paradigm accounts for drift-free estimation also when slow or no motion at all is present: rather than using an optimization window of time-successive poses, our kept keyframes may be spaced arbitrarily far in time, keeping visual constraints, while still incorporating an IMU term. We provide a strictly probabilistic derivation of IMU error terms and the respective information matrix, relating successive image frames without explicitly introducing states at IMU rate.

2.7 Deep Learning

The research in deep neural network absolutely thrive on big data. This is promoted by the powerful computation unit GPU and large scale dataset. It fills in gaps that there should have a complete problem formulation. This deep neural network works in an end-to-end manner by learning a mapping between the input data and the target label space. Give the training data x and corresponding label y, the training process is to find a optimal parameters that minimizes the loss between model prediction and label:

$$\omega^{\star} = \underset{\omega}{\operatorname{argmin}} \ell(f(x;\omega), y)$$

where ω is the parameters; $f(x, \omega)$ represent the output of the neural network given the input data x; ℓ stands for the loss function measuring the distance between the out and the label. The optimal parameters of the neural network are obtained by minimizing the loss.

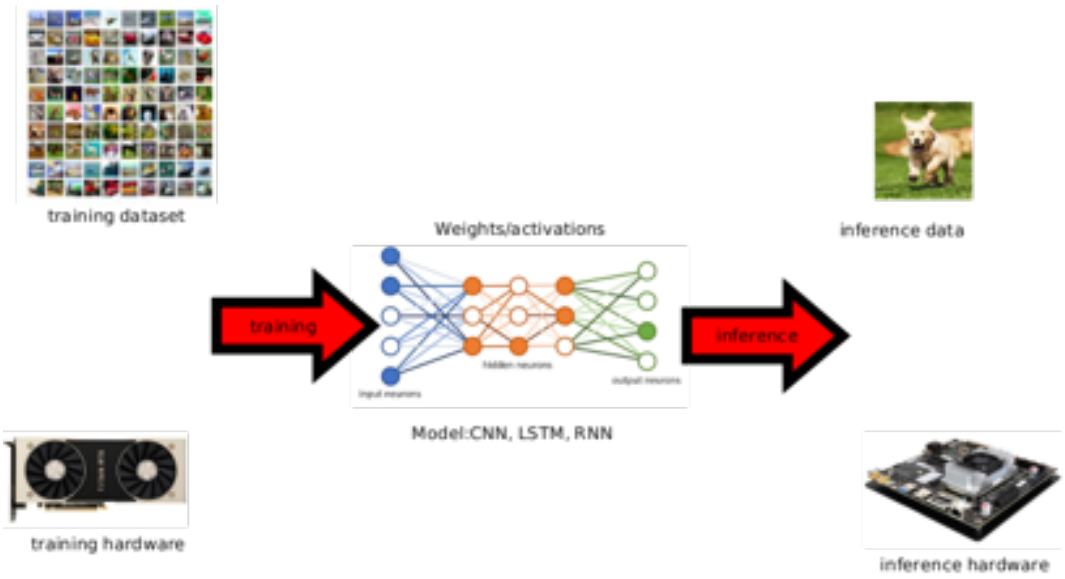


Figure 2.3: Illustration of different state estimation methods.

In practice, the parameters are iteratively updated based on a mini-batch of the training data. According to the rules of stochastic gradient decent (SGD) (66) or other solvers, the parameters are updated in the gradient decent direction with the step size μ:

$$\omega := \omega - \mu \nabla f(x; \omega)$$

After training, the model is able to directly map the input data to the output space $f(x; \omega^{\star})$.

2.7.1 Building Blocks of Deep Neural Network

Deep neural network is a generalization of the traditional neural network that explicitly transform the input to develop the output. Typically, the input to a neural network is an image, which has three dimensions: width, height and color channels (RGB), and the spatial information of neighboring pixels can be preserved. The deep neural network consists of lots of layers. Each layer is a type of

operator that take calculation on the input tensor. There are several basic layers in the neep neural network, and the fundamental one is the convolution layer.

- Convolution

The convolution layer takes a feature map as input, which is of width w, height h, and channel c.

- Pooling

Pooling is another important and unique operation in ConvNets, it allows small translation invariance and helps to reduce the size of the input feature map.

- Activation function

The activation function is a nonlinear function that promotes the deep neural network to approximate to complex nonlinear model.

Chapter 3

Direct Visual-Inertial Odometry with Semi-Dense Mapping

3.1 abstract

The paper presents a direct visual-inertial odometry system. In particular, a tightly coupled non-linear optimization based method is proposed by integrating the recent advances in direct dense tracking and Inertial Measurement Unit (IMU) pre-integration, and a factor graph optimization is adapted to estimate the pose of the camera and rebuild a semi-dense map. Two sliding windows are maintained in the proposed approach. The first one is based on Direct Sparse Odometry (DSO), which estimate the depths of the points from recent keyframes for mapping and dense visual tracking. In the second one, measurements from the IMU pre-integration and dense visual tracking are fused probabilistically using a tightly-coupled, optimization-based sensor fusion framework. As a result, the IMU pre-integration provides additional constraints to suppress the scale drift induced by the visual odometry. Evaluations on real-world benchmark datasets show that the proposed method achieves competitive results in indoor scenes.

3.2 Introduction

In the research area of robotics, camera motion estimation and 3D reconstruction take fundamental places in navigation and perception, such as unmanned aerial vehicle (UAV) navigation (78) and indoor reconstruction (29; 28). Among these applications, an robust camera motion tracking and 3D map of the environment are desired at the same time. Most existing methods formulate this problem as simultaneously localization and mapping (SLAM), which characterized on the sensors

it used. Recent efforts include visual SLAM and visual inertial navigation system (VINS).

Visual odometry (100) estimates the depth of features, based on which, track the pose of the camera. In contrast, direct visual odometry working directly on pixels without the feature extraction pipeline is free of the issues in feature based methods. This method is able to achieve drift-free estimation for slow motion. However, Direct visual odometry is also subject to failure as images can be severely blurred by changing illumination, and fast motion. Consequently, aggressive motion of the UAV (87; 86) with significant large angular velocities and linear accelerations makes the state estimation subject to scale drift immediately.

Pre-integrating IMU measurements can improve visual odometry remedy this issue by providing an additional short term measurements. IMU provides noisy but outlier free measurement with high frequency, making it an ideal device for fast motion tracking. We are convinced that associating the measurements of direct visual tracking and IMU pre-integration is able to achieve robust motion estimation within challenging environment.

This paper works on a tightly coupled fusion system for state estimation. We combine direct dense tracking and IMU pre-integration by adopting the concept of keyframes, and make a probabilistic derivation of the IMU errors and the corresponding information matrix. This paper proposes a robust and fully integrated system for direct visual inertial odometry. The novelties of the proposed system include: 1) The combination of the direct photometric information and the edge features, which are points with sufficiently high image gradient magnitude. We work with the intensity of pixels, and the system is more robust and reliable than other methods based on detected features. 2) IMU pre-integration. The IMU pre-integration provides scale information by integrating the IMU measurements. Benefiting from the use of a factor graph, tracking and mapping are focused in a local covisible area, which has a bounded size and is independent of the global map. 3) Tightly coupled optimization. The measurements of both the IMU pre-integration and the dense visual tracking are fused probabilistically within a single tightly-coupled, optimization-based framework. In this paper, the dense visual tracking results provide the visual constraints between current frame and the reference frame, while the IMU pre-integration provides

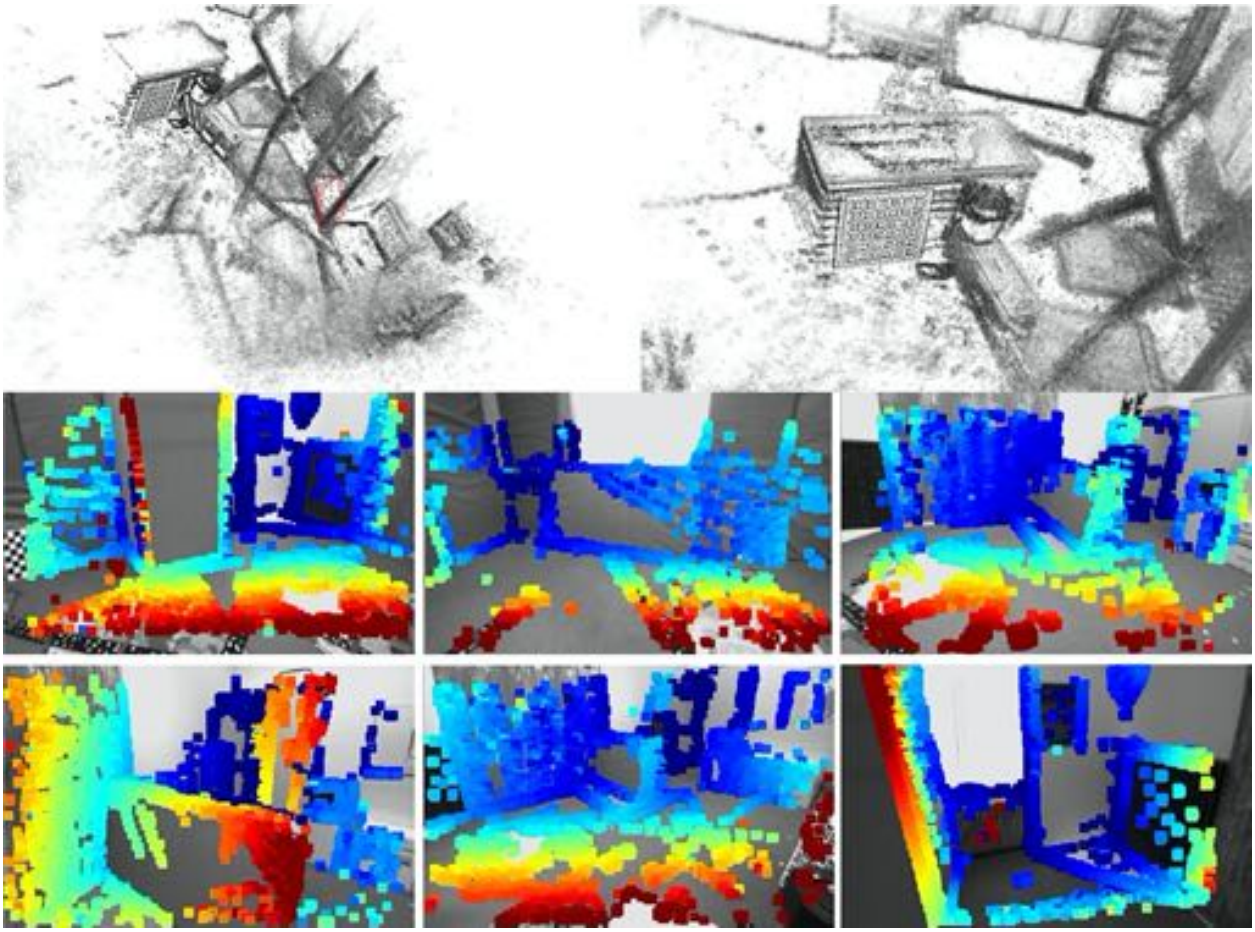


Figure 3.1: 3D reconstruction and depth estimation on EuRoC dataset. The first row shows the 3D reconstruction on the V1_easy sequence and the bottom rows show the depth maps for frame tracking.

constraints between the two consecutive frames.

In the remainder of the paper, we first review some related state-of-the-art works in Sect. 3.3, followed by the formulation of the problem in Sect. 3.4. In Sect. 3.5, an overview of the system is described, followed by the details of the IMU and visual measurement. Dense mapping is introduced in Sect. 3.6. Sect. 3.7 shows implement details. The experimental results and evaluations are discussed in Sect. 3.8. At last, we make a conclusion for the paper in Sect. 3.9.

3.3 Related Work

Simultaneously localization and mapping (SLAM) has a long history in monocular scenarios, prominent examples include dense tracking and mapping (DTAM) (100), semi-direct visual odometry (SVO) (39) and large-scale direct (LSD) monocular SLAM (30), which work on sparse features and estimate the camera motion through a prediction and correction fashion. Direct methods work directly on image intensities and attract a lot of attention since they are robust and computational efficient without feature detection. It is demonstrated that direct methods are more suitable for dense mapping than feature based method, and when enhanced by edge alignment (70), they can deal with changing illumination and fast motion. More recently, direct sparse odometry (DSO) (28) presented an impressive semi-dense 3D reconstruction of the environment through a sliding window optimization. This direct method minimizes the photometric error of points with sufficiently high image gradient magnitude (edges) using a non-linear optimization framework.

More recently, people pay great attention to the 3D SLAM, like the semantic SLAM (172) and the deep learning enhanced SLAM (129). Semantic SLAM focuses on simultaneous 3D reconstruction and material recognition and segmentation, which ends up with a real-time end-to-end system. Deep learning is very promising technique in computer vision. (129) takes the advantage of depth prediction from convolutional neural networks (CNNs) to enhance the performance of monocular SLAM. DSO (28) utilizes one monocular camera for 3D reconstruction within an inverse depth estimation framework. It is based on photometric error optimization of windowed sparse bundle adjustment.

All of these methods try to explore the scene within the reconstructed 3D map, they suffer the problem of scale drift. There are two main approaches towards solving the scale drift problem with extra measurements: the batch nonlinear optimization method and the recursive filtering method. The former one jointly minimizes the error originated from the IMU pre-integration and the photometric difference of the visual alignment (78); while the recursive algorithms usually utilize separately steps to estimate the state. There are two main lines to fuse these two kinds of measurements: loosely-coupled fusion (118; 141; 109) and tightly-coupled fusion (78; 86; 21; 133).

In loosely-coupled fusion, a filtering framework (118; 81; 80) is usually used. The optimization problem is separated into two sub-problems: a prediction step of the visual measurements to estimate state and a update step of the IMU measurements to correct the state (118; 81). Visual measurements of direct dense tracking are first performed for high-level pose information, then the predicted pose is updated with the inertial measurements. This filtering frame leads to sub-optimal solution with lower computational cost. In contrast, a single framework is utilized for tightly-coupled fusion, which formulates the fusion of visual and inertial measurements as an optimization problem. Linearizion technology is adopted to convert the non-linear representation of Lie group into a graph optimization problem. Tightly-coupled fusion achieves high estimation accuracy with higher computational cost.

3.4 Problem Formulation

To estimate the trajectory with the semi-dense map. we maintain a local window that contains one keyframe and two latest regular frames. Based on the point cloud of the DSO, a visual inertial fusion system are proposed with the IMU pre-integration constraint and the frame tracking constraint. Figure. 3.2 shows the graph representation. The details of the proposed system will be illustrated in this section.

3.4.1 Notation

We use the following notation in the paper:

- $\hat{m}_{k+1}^{k}$: the IMU measurements between the time interval of the k-th and the $(k+1)$-th images;
- $\hat{m}_{c}^{ref(c)}$: the dense tracking result of image c with respect to its corresponding reference frame $ref(c)$;
- $r_{IMU}(\hat{m}_{k+1}^{k}, \pi_{k+1}^{k})$: the residual between the IMU pre-integration and the state π_{k+1}^{k};
- $r_{I}(\hat{m}_{c}^{ref(c)}, \pi_{c}^{ref(c)})$: the dense tracking residual between $\hat{m}_{c}^{ref(c)}$ and the state $\pi_{c}^{ref(c)}$;

- Σ_{IMU}: the associated covariances of the IMU measurements;
- Σ_I: the associated covariances of the image alignment;
- π_k^G: the k-th state under the global coordinate system G;
- p_k^G: represents the k-th position state in the local window;
- v_k^G: represents the k-th velocity state in the local window;
- θ_k^G: represents the k-th angular state in the local window;
- b_a: is the state representing the k-th accelerometer bias in the local window;
- b_g: is the state representing the k-th gyro bias in the local window;
- $\theta_k^G = log(R_k^G)^\vee$.

Our system maintains a local window containing several states. One state is defined in the global coordinate system as π_k^G including the position, velocity, orientation, accelerometer bias, and gyroscope bias. The full state space π is defined as:

$$\pi_k^G = \begin{bmatrix} p_k^G & v_k^G & \theta_k^G & b_a & b_g \end{bmatrix} \tag{3.1}$$

$$\pi = \begin{bmatrix} \pi_0^G & \cdots & \pi_k^G & \cdots & \pi_n^G \end{bmatrix} \tag{3.2}$$

3.5 Visual-Inertial Odometry

This section presents an overview of the visual inertial system and the method of combining the direct visual measurements and inertial measurements. In visual odometry or visual SLAM, the camera pose estimation is formulated as a nonlinear optimization problem, which minimize the reprojection error of the landmarks observed by the camera. Unlike in work (86), which utilizes stereo cameras for depth estimation, we adopt the DSO to build a semi-dense map and take the depth estimation for camera motion tracking. Figure 3.2 shows the factor graph, where the visual

inertial fusion takes the points in local map as landmarks which provides a set of geometric constraints to related the observation states. The local map points are a collection of points with depth information in the latest keyframes.

The proposed method formulates the visual-inertial odometry problem as one joint optimization. The solution is obtained by minimizing a loss function $J(\pi)$ consisting of both the weighted landmarks reprojection error r_I and the weighted IMU integrating error r_{IMU}.

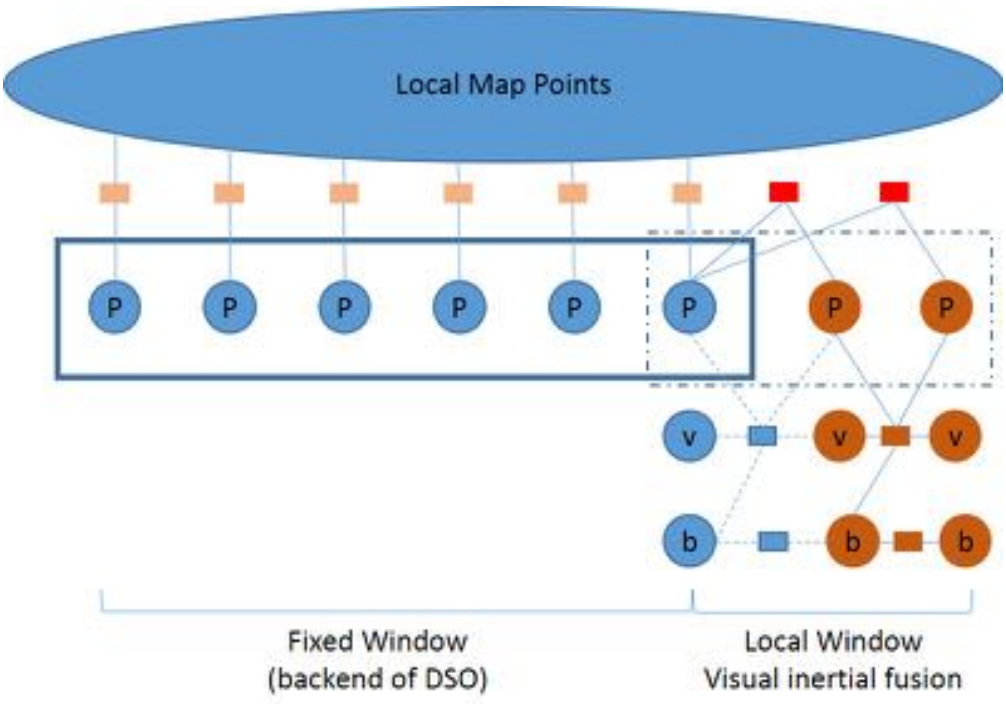


Figure 3.2: Overview of the system. After the fixed window used by DSO, we maintain a local window that contains one keyframe and two latest regular frames. Based on the point cloud of the DSO, a visual inertial fusion system are proposed with the IMU pre-integration constraint and the frame tracking constraint. P stands for pose states. v stands for the velocity states. b stands for the bias states. Square represents the states and rectangle represents constraints between states.

$$
\begin{aligned}
J(\pi) := & \sum_{k \in S_{IMU}} \| r_{IMU} \left(\hat{m}_{k+1}^{k}, \pi_{k+1}^{k} \right) \|_{\Sigma_{IMU}}^{2} \\
& + \sum_{c \in C_I} \| r_I \left(\hat{m}_{c}^{ref(c)}, \pi_{c}^{ref(c)} \right) \|_{\Sigma_I}^{2}
\end{aligned}
\tag{3.3}
$$

From the probabilistic view (37), a factor graph takes the available measurements and infers

the posterior probability of the variables:

$$\begin{aligned} p(\pi_k|\mathscr{M}_k) &\propto p(\pi_0)p(\mathscr{M}_k|\pi_k) \\ &= p(\pi_0)\prod_{(i,j)\in\mathscr{K}_k} p(\mathscr{M}_I,\mathscr{M}_{IMU}|\pi_k) \\ &= p(\pi_0)\prod_{k\in\mathscr{S}_{IMU}} p(\mathscr{M}_{IMU}|\pi_k)\prod_{c\in\mathscr{C}_I} p(\mathscr{M}_I|\pi_k) \end{aligned} \tag{3.4}$$

We assume all the distributions to be Gaussian. The MAP estimate corresponds to the minimum of the negative log-posterior. The negative log-posterior can be written as a sum of the squared residual errors:

$$\begin{aligned} \mathscr{X}_k^\star &= \underset{\pi_k}{\operatorname{argmin}} \; -\log_e p(\pi_k|\mathscr{M}_k) \\ &= \underset{\pi_k}{\operatorname{argmin}} \|r_0\|^2_{\Sigma_0} + \sum_{k\in\mathscr{S}_{IMU}} \|r_{IMU}\|^2_{\Sigma_{IMU}} + \sum_{c\in\mathscr{C}_I} \|r_I\|^2_{\Sigma_I} \end{aligned} \tag{3.5}$$

Without loss of generality, we optimize (3.5) iteratively using the Gaussian-Newton method by combining the results from the IMU pre-integration (Sect. 3.5.1) and the direct dense tracking (Sect. 3.5.2), and ignoring the prior term. To show the procedure of this processing, we reduce the terms and reformulate the cost function as:

$$F(x) = \sum r(x)^2 \tag{3.6}$$

To solve this minimization problem, we linearize the equation with Jacobian J_m and add a penalty term W, which are the inverse of Σ, to the equation. After this, the cost function is vectorized as:

$$\begin{aligned} F(\hat{x}+\Delta x) &= r(\hat{x}+\Delta x)^T W r(\hat{x}+\Delta x) \\ &= (r+J_m\Delta x)^T W (r+J_m\Delta x) \\ &= r^T W r + 2r^T W J_m \Delta x + \Delta x^T J_m^T W J_m \Delta x \end{aligned} \tag{3.7}$$

Following some basic operations, the solution of the minimization is obtained as:

$$\begin{aligned} J_m^T W J_m \Delta x &= -J_m^T W r \\ H\Delta x &= -b \\ \Delta x &= -H^{-1} b \end{aligned} \tag{3.8}$$

3.5.1 IMU Measurement

Usually, the frequency of the IMU is higher than that of the camera. There are tens of IMU measurement between the two consecutively image frames. The IMU pre-integration between two images provides a prior for image alignment, and also works as a different kind of constraint within the factor graph. The pre-integration $\hat{m}_{k+1}^k$ is given by:

$$\hat{m}_{k+1}^k = \begin{bmatrix} \hat{p}_{k+1}^k \\ \hat{v}_{k+1}^k \\ \hat{\theta}_{k+1}^k \\ \hat{b}_a^{k+1} \\ \hat{b}_g^{k+1} \end{bmatrix} = \begin{bmatrix} \sum_{i=k}^{k+1} \{ \frac{1}{2} \hat{R}_i^k (\hat{a}_i^i + b_a^i - R_G^i g^G) dt^2 + \hat{v}_i^k dt \} \\ \sum_{i=k}^{k+1} \hat{R}_i^k (\hat{a}_i^i + b_a^i - R_G^i g^G) dt \\ log(\Pi_{i=k}^{k+1} exp([\hat{\omega}_i^i + b_g^i]_\times dt))^\vee \\ \hat{b}_a^k + \eta_a^k dt \\ \hat{b}_g^k + \eta_g^k dt \end{bmatrix} \tag{3.9}$$

where g^G represents the gravity in the global coordinate system G, and $[\,]_\times$ represents the operator for skew-symmetric matrix. $\hat{a}_i^i$ denotes the linear acceleration measured by the IMU and $\hat{\omega}_i^i$ is the angular acceleration measured by IMU. η_a^k and η_g^k are white noise, which affect the accelerometer bias $\hat{b}_a^k$ and gyro bias $\hat{b}_g^k$. These biases are initialized to zeros at the beginning and updated at every optimization step. The updated values computed at each step are used for the next IMU pre-integration. According to the IMU propagation theory (86), the covariance is calculated according

to:

$$\Sigma_{IMU}^{k+1} = F_d(\hat{m}_{k+1}^k)\Sigma_{IMU}^k F_d^T(\hat{m}_{k+1}^k) + G(\hat{m}_{k+1}^k) Q G^T(\hat{m}_{k+1}^k) \tag{3.10}$$

where Q contains all the noise covariances, which are constants. $G(\hat{m}_{k+1}^k)$ denotes the noise transition matrix, and $F_d(\hat{m}_{k+1}^k)$ denotes the transition matrix of the discrete-time error state. In this work, it is defined as:

$$F_d(\hat{m}_{k+1}^k) = \begin{bmatrix} I & dt & -\frac{1}{2}\lfloor R_{k+1}^k(a+\hat{b}_g^{k+1})\rfloor_\times dt^2 & -\frac{1}{2}R_{k+1}^k dt^2 & 0 \\ 0 & I & -\frac{1}{2}\lfloor R_{k+1}^k(a+\hat{b}_g^{k+1})\rfloor_\times dt & -R_{k+1}^k dt & 0 \\ 0 & 0 & -R_k^{k+1} dt & 0 & -R_{k+1}^k dt \\ 0 & 0 & 0 & I & 0 \\ 0 & 0 & 0 & 0 & I \end{bmatrix} \tag{3.11}$$

Replacing the terms and reforming the equation, the residual between the IMU pre-integration and the states can be calculated by the following equation:

$$r_{IMU}(\hat{m}_{k+1}^k, \pi_{k+1}^k) = \begin{bmatrix} R_G^k(p_{k+1}^G - p_k^G - v_k^G dt) \\ R_G^k(v_{k+1}^G - v_k^G) \\ R_G^k R_{k+1}^G \\ b_a^{k+1} \\ b_g^{k+1} \end{bmatrix} \ominus \begin{bmatrix} \hat{p}_{k+1}^k \\ \hat{v}_{k+1}^k \\ \hat{R}_{k+1}^k \\ b_a^k \\ b_g^k \end{bmatrix} \tag{3.12}$$

We assume that:

$$R_G^k R_{k+1}^G \ominus \hat{R}_{k+1}^k = log((\hat{R}_{k+1}^k)^T R_G^k R_{k+1}^G)^\vee$$

and according to the infinitesimal increments in $SO(3)$ (21; 149), the Jacobian of the residual with respect to the error state is formulated as:

$$
\begin{aligned}
J_{IMU} &= \begin{bmatrix} \dfrac{\partial r_{IMU}(\hat{m}_{k+1}^{k}, \pi_{k+1}^{k})}{\partial \delta \pi_{k}^{G}} & \dfrac{\partial r_{IMU}(\hat{m}_{k+1}^{k}, \pi_{k+1}^{k})}{\partial \delta \pi_{k+1}^{G}} \end{bmatrix} \\
&= \begin{bmatrix}
-R_G^k & -R_G^k \lfloor R_G^k (p_{k+1}^G - p_k^G) \rfloor_\times & \dfrac{\partial r_{\Delta p_k^G}}{\partial \delta b_a^k} & \dfrac{\partial r_{\Delta p_k^G}}{\partial \delta b_g^k} & R_G^k & 0 & 0 & 0 \\
0 & -R_G^k \lfloor R_G^k (v_{k+1}^G - v_k^G) \rfloor_\times & \dfrac{\partial r_{\Delta v_k^G}}{\partial \delta b_a^k} & \dfrac{\partial r_{\Delta v_k^G}}{\partial \delta b_g^k} & 0 & R_G^k & 0 & 0 \\
0 & 0 & -R_G^{k+1} R_k^G & \dfrac{\partial r_{\Delta R_k^G}}{\partial \delta b_a^k} & \dfrac{\partial r_{\Delta R_k^G}}{\partial \delta b_g^k} & 0 & 0 & I \\
0 & 0 & 0 & -I & 0 & 0 & I & 0 \\
0 & 0 & 0 & 0 & -I & 0 & 0 & I
\end{bmatrix}
\end{aligned}
\tag{3.13}
$$

The details of the calculation will be defined in the A.1.

3.5.2 Visual Measurement

Between the time interval of two consecutive images, There could have several IMU measurements. We pre-integrate these IMU measurements, and once this pre-integration is done, we separate the image frames into two categories. A new frame is categorized as a *key frame* if its overlap with the current reference frame is less than a predefined threshold or the estimated distance between the two images is bigger than a predefined threshold. The first frame is automatically set as the key frame and other frames are considered as *regular frames*. A key frame maintains a 3D map (or point clouds) of the environment and works as a reference to track the subsequent frames. With one monocular camera, the inverse depth estimation is adapted for point tracking, and the depth of the points in the new key frame is initialized from the estimation of DSO.

This system allows the subsequent frames to have small movement in relation to the latest reference. To get the infinitesimal motion increment from the the current frame to the key frame,

we minimize the photometric error iteratively, which is the sum of the intensity differences r_{ij} of all pixels in that frame.

$$\hat{m}_c^{ref(c)} = \begin{bmatrix} \hat{p}_c^{ref(c)} \\ 0 \\ \hat{R}_c^{ref(c)} \\ 0 \\ 0 \end{bmatrix} = \arg\min_{p_c, R_c} \sum_i \sum_j r_{ij} \left(\hat{m}_c^{ref(c)} \right)^2 \tag{3.14}$$

$$r_{ij} \left(\hat{m}_c^{ref(c)} \right) = I_{ref(c)}(u_{ij}) - I_c(w(R^c_{ref(c)} w^{-1}(u_{ij}, d_u) + p^c_{ref(c)})) \tag{3.15}$$

where $I(u_{ij})$ represents the intensity of the pixel at u_{ij}, and d_u denotes the associated depth. $w(\ )$ is a map function that project the point in 3D space onto the image plane, with $w^{-1}(\ )$ the inverse of the projection function.

Once the measurement $\hat{m}_c^{ref(c)}$ in the reference frame's coordinate system is available, we can compute the residuals between the current states and the direct dense tracking results according to the following equation:

$$r_I \left(\hat{m}_c^{ref(c)}, \pi_c^{ref(c)} \right) = \begin{bmatrix} R_G^{ref(c)} (p^G_{ref(c)} - p^G_c) \\ 0 \\ R_G^{ref(c)} R_c^G \\ 0 \\ 0 \end{bmatrix} \ominus \hat{m}_c^{ref(c)} \tag{3.16}$$

and the Jacobian matrix of this dense tracking residual with respect to the 15 states is a sparse

matrix:

$$
\begin{aligned}
J_I &= \begin{bmatrix} \dfrac{\partial r_I\left(\hat{m}_c^{ref(c)}, \pi_c^{ref(c)}\right)}{\partial \delta \pi_c^G} & \dfrac{\partial r_I\left(\hat{m}_c^{ref(c)}, \pi_c^{ref(c)}\right)}{\partial \delta \pi_{ref(c)}^G} \end{bmatrix} \\
&= \begin{bmatrix} -R_G^{ref(c)} 0000 R_G^{ref(c)} 0 \lfloor R_G^{ref(c)}(p_{ref(c)}^G - p_c^G) \rfloor_\times 00 \\ 0 \quad 0000 \quad 0 \quad 0 \quad 0 \quad 00 \\ 0 \quad 0I00 \quad 0 \quad 0 \quad -R_G^c R_{ref(c)}^G \quad 00 \\ 0 \quad 0000 \quad 0 \quad 0 \quad 0 \quad 00 \\ 0 \quad 0000 \quad 0 \quad 0 \quad 0 \quad 00 \end{bmatrix}
\end{aligned} \tag{3.18}
$$

3.6 Semi-Dense Mapping

Fast camera motion tracking and 3D reconstruction require high update rate to successfully track the camera. According to (28), we maintain a semi-dense map of the pixels with high gradient to reduce the computational cost and ensure the tracking accuracy. The selected pixels are considered as point candidates, which will be tracked in the subsequent frames and used for the photometric error r_{ij}. To reduce the tracking error, We attempt to identify the potential outliers as early as possible and remove those when the photometric error surpass a threshold. For the best tracked point candidates, we follow the tracking strategy from LSD-SLAM (31) to compute the depth $\hat{d}_u$. The points associated with the depth serve as landmarks for camera tracking. The computed depth is used to define the search interval during the subsequent tracking.

$$
\hat{d}_u = \underset{d_u}{\operatorname{argmin}} \sum_i \sum_j r_{ij}^2 \tag{3.19}
$$

$$
r_{ij} = I_{ref(c)}(u_{ij}) - I_c(w(R_{ref(c)}^c w^{-1}(u_{ij}, d_u) + p_{ref(c)}^c)) \tag{3.20}
$$

The terms in Eq. 3.20 are the same as those in Eq. 3.15. When a new keyframe is added

according to the tracking performance, we project all active points onto it and establish a semi-dense depth map in the local coordinate system. The candidate points among all keyframes in the local window are added to the optimization window.

Figure 3.3 illustrates the pipeline of the proposed visual inertial odometry system. Between the time interval of two consecutive images, the output data from the IMU sensor is first pre-integrated according to Eq. 3.9. In the front-end, the direct dense tracking thread calculates the incremental camera motion using the direct keyframe-to-frame dense tracking algorithm, Eq. 3.14, proceeded by IMU pre-integration. Based on the distance between the keyframe and the latest frame and the current tracking performance, it is determined whether to add this frame as a keyframe or regular frame into the local window. The distance is a weighted combination of the translation and rotation between the latest keyframe and the latest frame. If it is considered as a keyframe, a depth map for the candidate points will be generated from the DSO. If the tracking algorithm performs worse, the system reports a tracking failure. The back-end periodically checks the frames in the local window. The new frame and IMU pre-integration are added to the optimization thread. Graph optimization (Eq. 3.5) takes constraints from IMU pre-integration (Eq. 3.12), dense tracking (Eq. 3.16) and the prior (Eq. 3.22) to find the best estimation of all states within the local window. After adding new states and obtaining the estimation, we usually try to remove states with less information out of the local window for the purpose of saving computational power and maximizing the information stored in the local window. For this purpose, A two-way marginalization scheme is applied to remove the redundant states, which is describe in Sect. 3.7.3.

3.7 Robust Method

3.7.1 Robust Norm

Image processing usually suffers from outliers due to image noise and ambiguity. L_2 norm is sensitive to outliers. Very small number of outliers may drive the solution away. The Huber function (33) is less sensitive to outliers. It is smooth near zero residual and affect large residuals

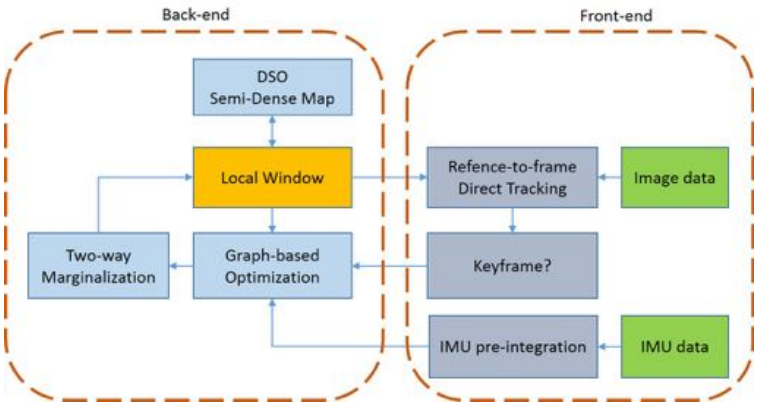


Figure 3.3: The pipeline of the proposed visual inertial odometry system, which comprises of the front-end for direct dense tracking and the back-end for optimization.

linearly. The Huber weight function is defined as:

$$\omega_{\text{Huber}}(\mathrm{x}) = \begin{cases} 1 & if\ |x| \leq 0; \\ \dfrac{k}{|x|} & otherwise. \end{cases} \tag{3.21}$$

where x is the residual and k is a given threshold. After obtaining the IMU residuals or the visual residuals, we apply the Huber weight function to the residuals aiming to reduce the effect of the outliers.

3.7.2 Keyframe and Point Management

The front-end of visual odomerty needs to determine the active frame/point set, provide initialization for new parameters and decide when a frame/point should be marginalized (28) so as to make the system computationally efficient and accurate. Our keyframe and point management is largely based on the DSO (28) except for that we add two more continuous regular frames into the sliding window (113). The two regular frames are connected by the IMU pre-integration.

3.7.3 Two-Way Marginalization

In the processing, the state set increases when a new frame is added, which requires more memory and computational resources. In order to increase its efficiency, we seek to remove frames with less information out of the local window based on a two-way marginalization scheme (78) and only optimize a certain number of states for real time performance. This scheme converts the constraints within the marginalized states into a prior, instead of simply dropping non-keyframes. Two-way marginalization scheme consists of two marginalization strategies to remove the frame in the local window. One is the front marginalization strategy. It drops the second newest frame in the local window since the second newest state is close to the reference frame within the local window. This strategy ensures that the time interval for each IMU pre-integration is bounded so as to reduce the accumulated IMU pre-integration error. The second newest state will be marginalized out of the local window at the next optimization step if the dense tracking performance well. The other one is the back marginalization strategy. It removes the oldest keyframe in the local window if the latest frame is added into the local window as a keyframe. The oldest state in the local window always provide constraints on the dense image alignment and the IMU pre-integration, while these constraints degraded by the cumulated error. In this situation, the oldest keyframe provides very few information. Marginalizing the oldest state into a prior improves the effectiveness of multi-constrained factor graph optimization.

For the prior matrix $\{\Lambda_p, b_p\}$, which is utilized to keep the constraints of the marginalized states. When the marginalized states are removed out of the local window, their information is transformed into the prior according to the Schur Complement method (118):

$$\begin{aligned}\Lambda_p = \Lambda_p &+ \sum_{k \in \mathscr{S}_{IMU}^{-}} (J_{k+1}^{k})^T (\Sigma_{k+1}^{k})^{-1} J_{k+1}^{k} \\ &+ \sum_{c \in \mathscr{C}_I^{-}} (J_I^{ref(c)})^T (\Sigma_I^{ref(c)})^{-1} J_I^{ref(c)}\end{aligned} \tag{3.22}$$

where $\mathscr{S}_{IMU}^{-}$ and $\mathscr{C}_I^{-}$ represent the set of removed IMU and visual measurement, respectively.

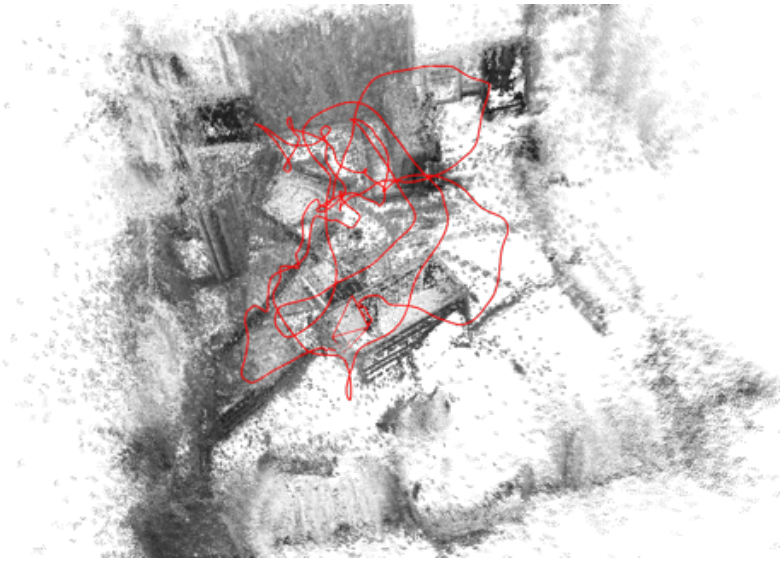


Figure 3.4: Overview of the 3D reconstruction on $V1\_01$ sequence. The red line is the estimated trajectory while the structure in black is the 3D point cloud.

3.8 Experimental Evaluation

In the literature of visual odometry, a bunch of motion tracking algorithms has been proposed. It could be visual only odometry, loosely coupled fusion, or tightly coupled fusion. How these methods perform in relation to each other is not studied adequately as the performances are typically based on different datasets charactered with different motion speeds and lighting conditions. We will thus compare the proposed method with the state-of-the-art odometry methods within one common dataset for a strong demonstration.

3.8.1 Experiment Setup

We take the European Robotics Challenge (EuRoC) dataset (12). It provides the ground truth at $1mm$ accuracy and contains 11 visual-inertial sequences recorded in 3 different indoor environment. The datasets are increasingly difficult to process in terms of flight dynamics and lighting conditions. The two sequences that we use are the V1_01 and the MH_02, which could be successfully handled and compared with other methods, and they are representative sequences of the dataset. The entire visual inertial odometry system is developed within ROS. It is noted that the

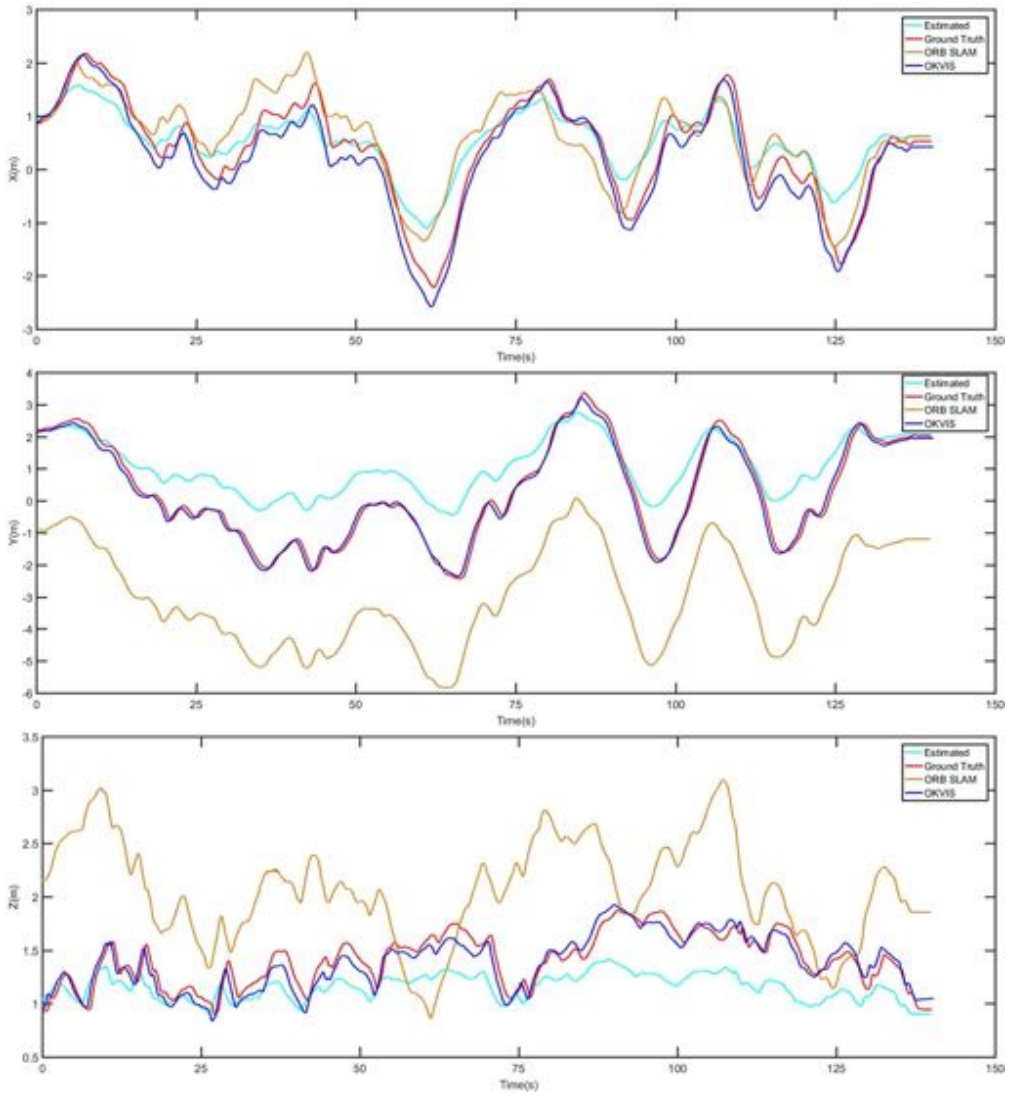


Figure 3.5: Comparison of trajectory estimations between our algorithm, the ground truth for the sequence *V*1_01, OKVIS, and ORB SLAM.

random walk of the IMU will induce systematic errors and make the system unstable without proper method to handle it. Unlike the work by (86), which is for 9 states estimation with the IMU fixed carefully to the global coordinate system, the coordinate system of the IMU is not aligned with the global coordinate system within the EuRoC dataset. We thus add another 6 states of IMU biases into the state space and update the states within the extended state space to ensure the success of state estimation.

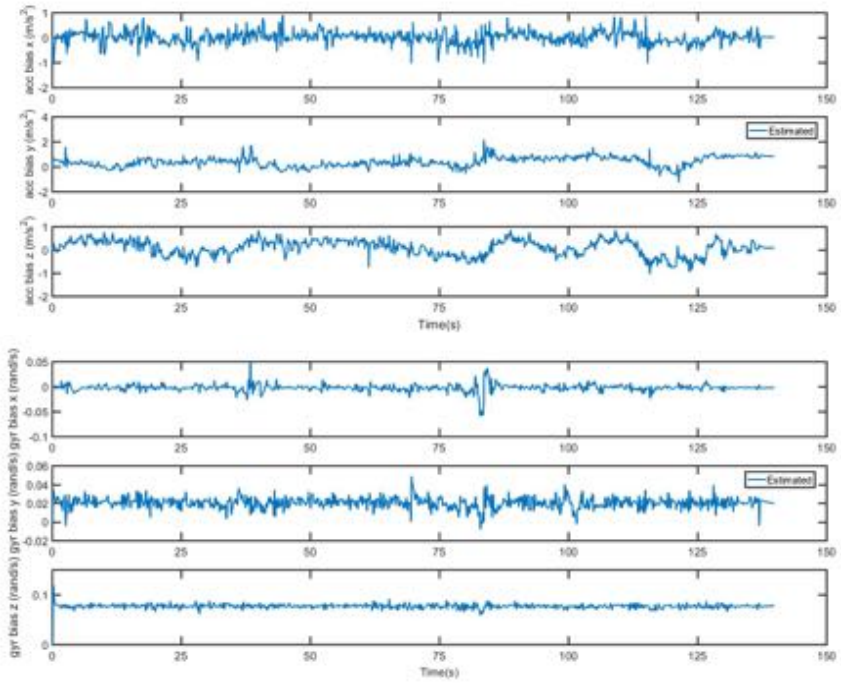


Figure 3.6: IMU acceleration bias and gyroscope bias estimation results from our algorithm.

3.8.2 Evaluation on EuRoC Dataset

In order to evaluate the performance of the developed visual inertial odometry system, two different methods dealing with the odometry problem are used for comparison. The OKVIS(78) fuses the IMU measurement and the visual measurement in a tightly coupled frame. The graph optimization guarantees its state-of-the-art performance. ORB-SLAM (99) performance motion tracking for visual odometry, which is charactered with the novel ORB features. Both methods create a sparse 3D map.

To ensure the validation, we conduct the comparison in the coordinate systems of the ground truth. The estimated pose from three systems are first recoded in their own coordinate system and them transformed to the target coordinate system. The comparisons of estimated trajectories from different methods are shown in Figure 3.5 and Figure 3.8.

Besides the pose, Our system create a semi-dense 3D map simultaneously. the reconstructed 3D map of the surroundings in the dataset are shown in Figure 3.4 and Figure 3.7, respectively.

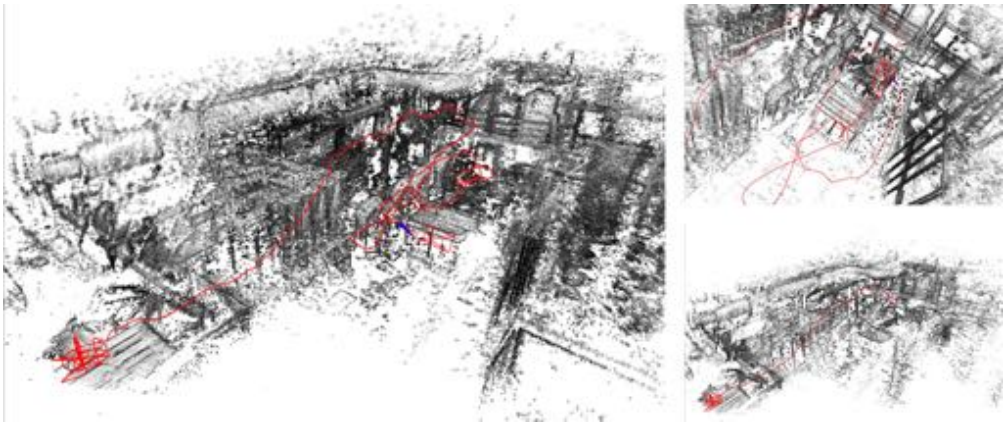


Figure 3.7: Whole view of 3D reconstruction on MH_02 sequence. The red line is the estimated trajectory while the black part is the 3D point cloud. The left part shows the reconstruction of the machine house. The right two images show the details from different view angular.

Figure 3.5 shows the comparison of trajectory estimations tested on the $V1\_01$ sequence. The dataset is characterized by fast motion and image blur. The ORB-SLAM runs for visual odometry without IMU measurements to provide information on short term movement. This results to the scale drift in three directions. OKVIS preforms the best (the closest to the ground truth) since it fuses the IMU measurement with the motion tracking results and includes the states on features into the state space. These feature states put much more constraints during the optimization and guarantee a globally optimal solution. Figure 3.6 shows the estimated biases for the IMU. In the proposed method, We perform a semi-dense reconstruction and do not take the estimated depth into the optimization formulation. And due to the depth estimation error from the DSO, the visual inertial fusion for motion tracking could lead to a reduced performance. The trajectory estimation results on MH_02 sequence are shown in Figure 3.8. The proposed approach and OKVIS, with IMU measurement, are able to track these sequences successfully. Nevertheless, ORB-SLAM fails to track in the two sequences in terms of robustness and accuracy.

Benefiting from the semi-dense tracking, the proposed method reconstructs a semi-dense 3D map simultaneously. Figure 3.7 presents the reconstruction of a machine house from the MH_02 sequence which is an indoor dark scene, and Figure 3.4 demonstrates the reconstruction of the environment in a small room from $V1\_01$ sequence. The ORB slam and OKVIS work on sparse

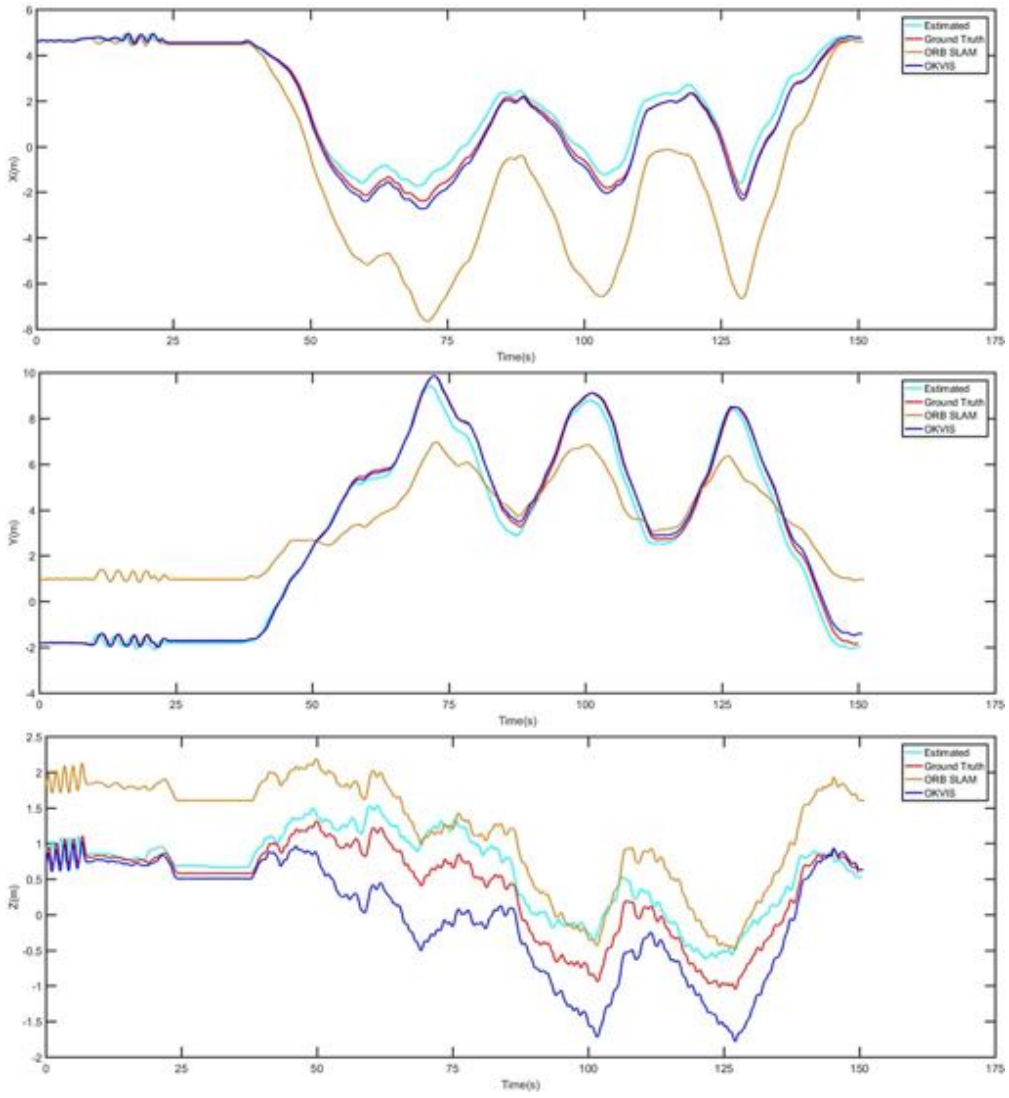


Figure 3.8: Comparison of trajectory estimations between our algorithm, the ground truth for the sequence *MH*_02, OKVIS, and ORB SLAM.

features which are a small portion of points. They can not generate a recognizable map of the environment.

From these challenging experiments, it demonstrates that the proposed system works with fast motion and different lighting condition, but it is still subject to scale drift with respect to the ground truth. In all cases, our inertial measurement from the IMU provides constraints to largely eliminate the drifts.

3.9 Conclusion

In this paper, we have proposed a direct visual-inertial odometry system that tracks camera motion and reconstructs 3D map of the environment simultaneously. This system integrates the direct dense image tracking and the IMU pre-integration into one formulation, minimizing a joint cost in a tightly coupled manner. The system maintains two sliding windows and probabilistically integrates the IMU pre-integration and the direct visual tracking results. The proposed system effective alleviates the scale drift issue of the visual odemetry. It has been evaluated on the popular dataset and compared with the state-of-the-art odometry methods. Benefiting from the optimization, the proposed system performs competitively state estimation.

Chapter 4

Attention-aware Neural Network for Camera Localization

4.1 Abstract

Despite the fact that various learning based navigation systems have been developed,few of them can capture the long-term temporal dependencies appropriately and select the relevant driving series to make predictions. We present a hybrid deep learning method for camera relocalization. The proposed system leverages the discriminative deep image representation from a convolutional neural networks to capture the long-term temporal dependencies appropriately and select the relevant driving series to make predictions of the six degree of freedom (6DoF) camera pose in an end-to-end fashion. Specifically, we formulate a novel spatial Attention CNN model for joint learning of soft pixel attention and hard graph constraints with simultaneous optimisation of feature representations that reduce the uncertainty at the feature extraction. This is dedicated to localize the discriminative parts which generates high-quality 6-degree pose estimation. The results show that compared to the state-of-the-art camera relocalization methods, our model produces comparable localization estimation and improves the system efficiency, without loss of accuracy.

4.2 Introduction

Deep learning method is becoming dominated in solving the computer vision relation applications, such as. This is largely due to the fact that deep neural network is able to extract more robust feature. Recently, the recurrent neural network and attention mechanism are employed to enhance the feature by establishing the long-term temporal dependency. While how dose this feature depen-

dency affects the performance in deep camera localization is remaining an open problem.

Being able to localize a vehicle or device by estimating a camera pose from an image is a fundamental requirement for many computer vision applications such as navigating autonomous vehicles, simultaneous localization and mapping (SLAM) (29), and Structure-from-Motion (SfM) (111). Most state-of-the-art approaches rely on local features such as SIFT (89) to solve the problem of image-based localization. These approaches are based on the low-level geometric features and do not work very well with environments with repeating structures and texture-less surfaces.

Traditionally, the camera localization problem is to employ image retrieval (108) techniques to identify the database photo most similar to the query. It apply standard techniques such as image descriptors, such as SIFT (89), fast spatial matching (104), bag of visual words (157) to find a representation of an unknown scene (a query image) in a database.

recently developed camera localization models rely on CNN (68) for extracting a variety of high-level knowledge from images and the combination of loss function between translation and rotation. while, they fail to get rid of the correlation between the background and salient object. And Their performance is significantly affected by this uncertainty. SalientDSO (83) focuses on generating saliency maps to understand the "attended" regions, This mechanism provides a more direct interpretation by making the attention a core part of the network.

In this paper, we aim to reduce the uncertainty of the deep structure that regresses the camera pose directly from a single RGB image. the related work is the MapNet (10), which consider to learn a constraint map, but this one is built upon to extract feature on the whole image.

The proposed system leverages the attention mechanism and long-short-term memory (LSTM) to capture the long-term temporal dependencies appropriately and select the relevant driving series to make predictions of the six degree of freedom (6DoF) camera pose in an end-to-end fashion. The Attention model enables us to build agents and classifiers that actively select important and task relevant information from visual inputs. To do this, the model generates attention maps, which can uncover some of underlying decision process used to solve the task. Once obtained the CNN feature, we propose to make use of graph constraints on the FC output, which performs structured

dimensionality reduction and chooses the most useful feature correlations for the task of pose estimation.

We observe that the attention mechanism focuses on the salient components of each input and the lstm establish the temporal dependency in the sequence of images. The fundamental feature of our approach is to unify the Attention mechanism and lstm in one framework for camera pose estimation. This allows us to retrieve a reliable geometry in both the feature and the estimated pose. The main contributions of this work are listed below.

- We formulate a novel idea of jointly learning attended feature and temporal dependency in the feature sequence. To our knowledge, this is the attempt of jointly deep learning multiple attention and temporal feature dependency for solving the camera pose estimation.
- We propose a attention mechanism to enhance the feature extraction in convolutional neural network. We learn soft pixel-level attention within arbitrary feature representations for maximising the correlated complementary information. This is achieved by devising a lightweight Harmonious Attention module capable of efficiently and effectively learning spatial and channel attention.
- The proposed model is qualitatively and quantitatively evaluated on multiple datasets. This contains the existing public dataset and UAV flight dataset. Its successful estimation reflect a superior performance over baseline models.

4.3 Related work

Camera localization. Camera localization is a significant technology in robotics that could be used for navigation, intelligent service and so on. Traditionally, the camera localization problem is to employ image retrieval (108) techniques to identify the database photo most similar to the query. It apply standard techniques such as image descriptors, such as SIFT (89), fast spatial matching (104), bag of visual words (157) to find a representation of an unknown scene (a query image) in a database. Visual odometry estimates the motion of a camera in real time using sequential

images (i.e., egomotion). This optimization based method, such as SLAM (149; 150), takes the advantage of computer vision techniques that seek for the geometry constrains. This approach is based on the low-level geometric features and do not work very well with environments with repeating structures and texture-less surfaces. They estimate the relative pose between consecutive images. However, the localization process itself is time-consuming in descriptor matching and the features are sensitive to the environment.

Deep Learning Based Localization. Recently proposed CNNs-based approaches have shown great success in image-based localization. Utilizing CNNs to directly regress camera relocalization was proposed by Kendall et al. (64). Their method, named PoseNet, adapts GoogLeNet (125) architecture pre-trained on large-scale image classification data to reconstruct 6-DoF camera pose from an RGB image. In the recent paper (63), Kendall et al. propose a novel loss function based on scene reprojection error and show its efficiency in appearance-based localization. The relative camera pose estimation (98; 76) introduce the relative pose constraint that optimize the relation between consecutive images. MapNet (10) combines both the absolute and relative pose constraints. LSTM-pose (135) employing the Recurrent Convolutional Neural Networks for camera relocalization. DeepVO (139) present an end-to-end framework for monocular visual odometry by using deep Recurrent Convolutional Neural Networks. VidLoc (19) introduce a deep spatio-temporal model for 6-DoF video-clip relocalization. VlocNet (134) integrates auxiliary learning to enhance visual localization and odometry. Object pose estimation (136; 147) is a closely related task that provide a different view of pose estimation. Our approach is built upon on these prior methods. We actively integrate these techniques for better performance and introduce the attention module to enhance the extracted CNN feature.

Attention Mechanism. Attention in deep neural network is a mechanism to assign different weights to different feature spatial regions depending on their feature content. It automatically predicts the weighted heat map to enhance the relevant features and block the irrelevant features during the training process for specific tasks. Intuitively, such weighted heat map could be applied to our purpose of pseudo-annotation generation. Attention mechanism has been proven to be beneficial

in many tasks such as image captioning (158), time sequence prediction (105), image-to-image translation (97), human pose estimation (16) and saliency detection (69). Thus weakly-supervised semantic segmentation (167) receives research attention to alleviate the workload of pixel-wise annotation for the training data. Harmonious attention convolutional neural network (82) simultaneously learns hard region-level and soft pixel-level attention within arbitrary person bounding boxes along with re-id feature representations for maximising the correlated complementary information between attention selection and feature discrimination for person re-identification.

Different from the previous works, we are the first to apply the attention mechanism to weakly supervised semantic segmentation to the best of our knowledge.

4.4 The proposed method

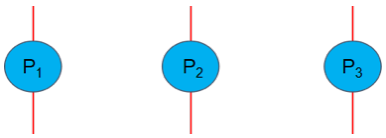
In this paper, we propose a geometry-attention dual-agent neural network for camera localization. The primary goal of our architecture is to precisely estimate the global pose by minimizing our proposed Geometric Loss function, which is promoted by constricting the search space in the self-attention weighted feature map.

Figure 2 illustrates our proposed model, which consists of feature extraction base, an attention network and a global pose regression head. Given a sequence of consecutive monocular images $\{I_t\}_{t=1}^{N}$, our network predicts the 6-DoF global pose $p_t = [x_t, w_t]$ in time t, where $x \in R^3$denotes the translation and $w \in R^3$ denotes the Euler angle.

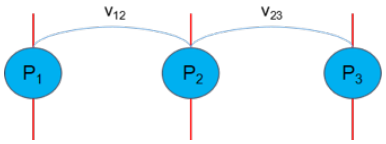
In the remainder of this section, we present the constituting parts of our architecture along with how the joint optimization is carried out.

4.4.1 Global Pose Regression

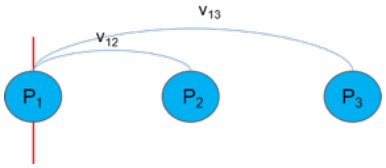
1) Geometric Consistency Loss: Learning both translational and rotational pose components with the same loss function is inherently challenging due to the difference in scale and units between both the quantities. It is easy to learn camera position in Euclidean space. However, learning



(a) PoseNet



(b) MapNet



(c) Ours

Figure 4.1: Diverse generation for the 'edge $\leftrightarrow$ photo' generation task. the first column represents the inputs, columns 3-4 show the generations in edge domain and columns 5-8 are generations in the photo domain.

orientation is more complex. Recently, two parametrization methods are common used: Euler angles and quaternions. The Euler angle representation suffers the problem of having multiple values representing the same angle, which causes them to be challenging to learn as a uni-modal scalar regression task. While the quadruple is an over parameterization of the 3-DoF rotation, and normalization of the output quadruple is required but often results in worse performance.

We describe the loss function for regressing the translational and rotational components in the Euclidean space.

$$\mathscr{L}_{adv} = \sum_{i=1}^{|\mathscr{D}|} h(p_i, p_i^{\star}) + \sum_{i,j=1, i\neq j}^{|\mathscr{D}|} h(v_{ij}, v_{ij}^{\star}) \tag{4.1}$$

where $v_{ij} = (x_i - x_j, w_i - w_j)$ is the relative camera pose between pose predictions p_i and p_j for images I_i and I_j . $h(\cdot)$ is a function to measure the distance between the predicted camera pose

p and the ground truth camera pose $p^\star$, defined as:

$$\begin{aligned} h(p, p^\star) = e^{-\beta}||x - x^\star||_1 + \beta \\ + e^{-\gamma}||w - w^\star||_1 + \gamma \end{aligned} \tag{4.2}$$

where β and γ are the weights that balance the translation loss and rotation loss. Both β and γ are learned during training with pre-defined initial value β_0 and γ_0 .

2) Graph Constraints: There are different ways to integrate the absolute pose estimation constraint and relative pose estimation constraints. Different models takes different strategies. The PoseNet only minimize the absolute pose constraint, while MapNet combines these two constraints. However, the relative pose estimation constraint used by MapNet takes the each two consecutive poses. In order to integrate motion specific features in our global pose regression network, we take the notion of keyframe into our method. To regress the 6-DoF relative pose from the images, we minimize the absolute pose error for the first keyframe. And the relative pose is calcutated between the keyframe and all other frames. This is illustrateds in Figure.

Similar to our approach used for the global pose regression, we additionally learn two weighting parameters to balance the loss between both components.

4.4.2 Spatial-and-Channel Attention Learning

In the literature, visual attention mechanism mainly focuses on the problem of identifying "where to look" on different visual tasks. As such, it can be naturally applied to camera localization, which is exactly to localize where to pay attention in. Based on the above consideration, we propose to set up a two streams attention mechanism. Assume the input to the attention module is a $3D$ tensor $x^l \in R^{c \times h \times w}$, where h, w, and c denote the number of pixel in the height, width, and channel dimensions respectively; and l indicates the level of this module in the entire network. The spatial-channel attention learning aims to produce a saliency weight map $A^l \in R^{c \times h \times w}$of the same size as x^l. Given the largely independent nature between spatial (inter-pixel) and channel (inter-scale)

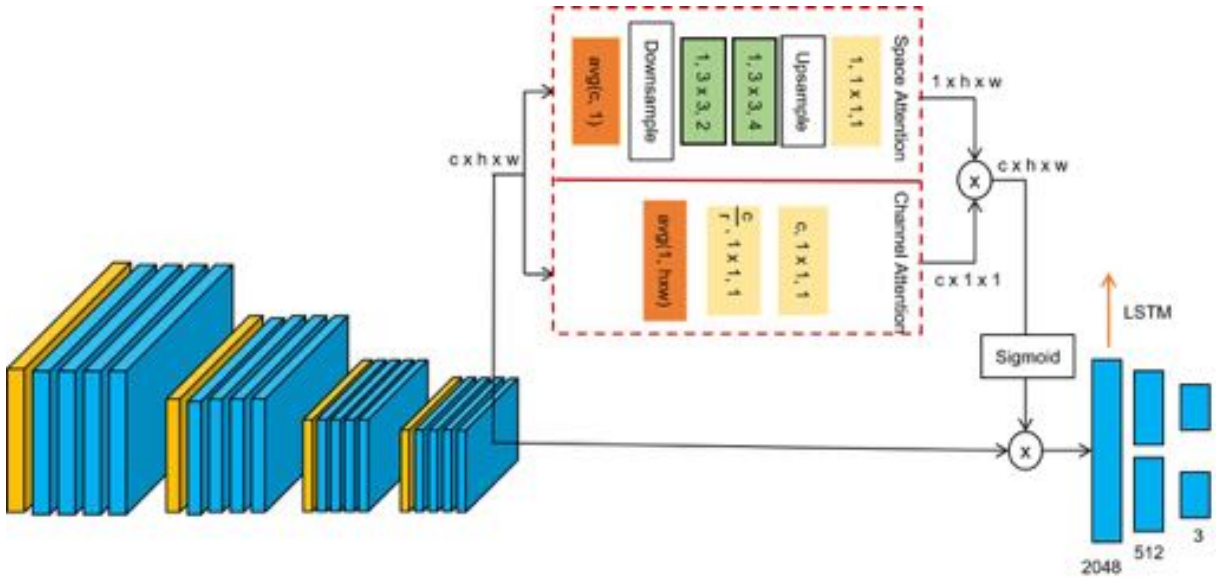


Figure 4.2: Diverse generation for the 'edge $\leftrightarrow$ photo' generation task. the first column represents the inputs, columns 3-4 show the generations in edge domain and columns 5-8 are generations in the photo domain.

attention, we propose to learn them in a joint but factorised way as:

$$A^l = S^l \times C^l \tag{4.3}$$

where $S^l \in R^{1\times h\times w}$ and $C^l \in R^{c\times 1\times 1}$ represent the spatial and channel attention maps, respectively. We perform the attention tensor factorisation by designing a two-branches unit (Fig. 3(a)): One branch to model the spatial attention S^l (shared across the channel dimension), and another branch to model the channel attention C^l (shared across both height and width dimensions). By this design, we can compute efficiently the full soft attention A^l from C^l and S^l with a tensor multiplication.

1) Spatial Attention The spatial attention module is a sub-network that consists of a channel-wise global averaging pooling layer, a conv layer of 3×3 filter with stride 2, two dilated conv layers, a resizing bilinear layer, and a scaling conv layer. In particular, the channel-wise global averaging pooling is defined as,

$$S^l = \frac{1}{c}\sum_{i=1}^{c} X^l_{i,1:h,1:w} \tag{4.4}$$

The channel-wise pooling is to reduce the input size of the following layers. The dilated conv layers is utilized especially to increase the receptive field. This learns one attention map shared by all channels. We finally add the scaling layer for automatically learning an adaptive fusion scale in order to optimally combining the channel attention.

2) Channel Attention. The Channel attention is modelled by a small squeeze-and-excitation sub-network. The average pooling layer first squeezes the spatial size to 1×1 for aggregating feature information distributed across the spatial space. This is described as

$$C^l = \frac{1}{h \times w} \sum_{i=1}^{h} \sum_{j=1}^{w} X^l_{i,j,1:c} \tag{4.5}$$

This operation generates a per-channel filter. Each channel represents information from the whole spatial tensor. This information is then used to establish the inter-channel dependency in the subsequent excitation operation and r represents the bottleneck reduction rate.

3) Blended Attention For facilitating the combination of the spatial and channel attention, we take a transformation to regularize the learned attention map. It is described as

$$A^l_{out} = T(A^l) \tag{4.6}$$

$T(\cdot)$ is a masking function based on a thresholding operation. In order to make it derivable, we use sigmoid function as an approximation

$$A^l_{out} = \frac{1}{1 + exp(-\omega(A^l - \sigma))} \tag{4.7}$$

where σ is the threshold matrix whose elements all equal to σ. ω is the scale parameter ensuring the elements in A^l_{out} approximately equal to 1 when A^l is larger than σ, or to 0 otherwise.

We then use the trainable attention map A^l_{out} to generate a soft mask to be applied on the original input image, obtaining $I \times A^l_{out}$. It represents the regions beyond the network's current attention.

4.4.3 Network Architecture

(I) Network Architecture We extend ResNet-34 (45) architecture with the proposed attention mechanism to predict the global pose. The architecture is comprised of four residual blocks with multiple residual units, where each unit has a bottleneck architecture consisting of three convolutional layers in the following order: 1×1 convolution, 3×3 convolution, 1×1 convolution. Each of the convolutions is followed by batch normalization, scale and Rectified Linear Unit (ReLU). As discussed in (134), Exponential Linear Units (ELUs) tend to reduce the bias shift in the network, in addition to avoiding the vanishing gradient and yield faster convergence. We replace the ReLUs in the residual block module with the ELUs. We also replace the last average pooling layer with global average pooling and subsequently add several inner-product layers, namely fc_1, fc_2 and fc_3. The first inner-product layer fc_1 is of dimension 2048 and the following two inner-product layers are of dimensions 3 and 4, for regressing the translation x and rotation w respectively. We take the advantage of recurrent units into our network. The LSTM is employed to exploit long term temporal dependency across the output of fc_1.

Our proposed Geometric Consistency Loss contains both the absolute pose error and the relative pose error. We feed one image sequence to the neural network at one training step. The model predicts the pose for each image in the sequence. According to the graph constraint, we calculate the relative pose for either the estimated pose or the ground truth pose. Then we measure the loss between the estimated one and the ground truth.

(I) Implementation Detail. In order to train our network on different datasets, we rescale the images maintaining the aspect ratio such that the shorter side is of length 256 pixels. We found that using random crops of 224×224 pixels acts as a better regularizer helping the network generalize better. For evaluations, we use the center crop of the images. We use the Adam solver for optimization with $\beta_1 = 0.5$,$\beta_2 = 0.999$. We train the network with an initial learning rate of $lr = 10^{-4}$ with a mini-batch size of 20 and a dropout probability of 0.5. Specifically $\beta = 0$ and $\gamma = -3$ are used for our Geometric Consistency Loss function.

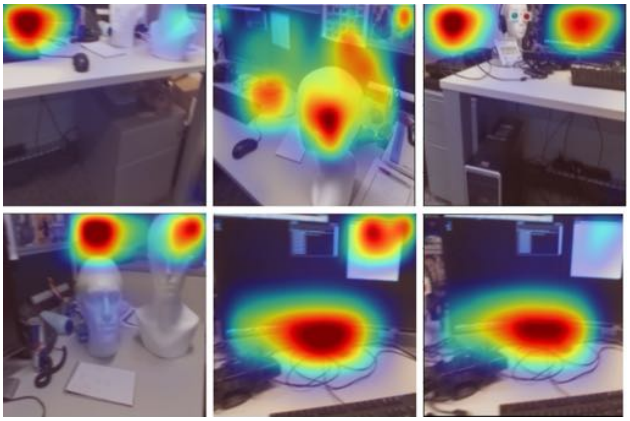


Figure 4.3: Visualisation of our attention learned in the 7 scene dataset. The discriminative regions of the images are highlighted

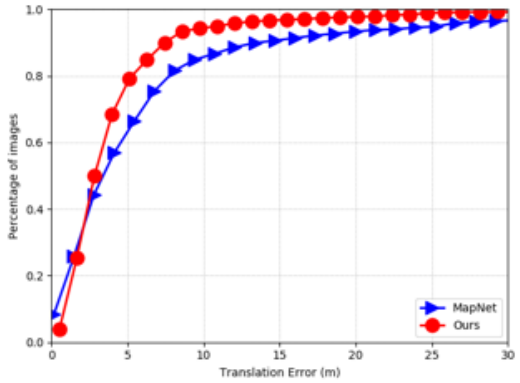


Figure 4.4: Cumulative distributions of the translation errors (m) for methods evaluated on Oxford RobotCar LOOP. $x-$axis is the translation error and $y-$axis is the percentage of frames with error less than the value.

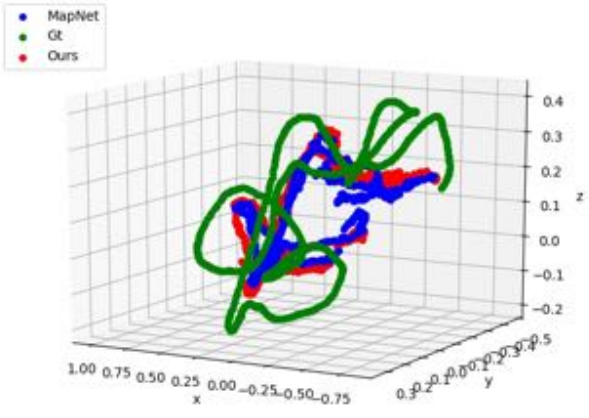


Figure 4.5: Camera localization results on the 7 scene dataset.

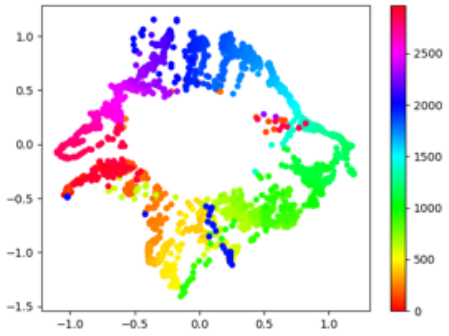


Figure 4.6: 2D multi-dimensional scaling (MDS) of the features from the second latest layer of the model trained on the LOOP sequence. The points are chronologically colored. We observed that the MDS created from the features of our model is close to be a rectangle that is consistent to the ground truth camera poses which is from a loop. This demonstrates the features extracted by our model is better correlated to the camera poses, hence improve the pose estimation accuracy.

4.5 Experimental Evaluation

In this section, we qualitatively and quantitatively evaluate our proposed method in comparison to the state-of-the-art on both indoor and outdoor datasets, followed by detailed analysis on the architectural decisions and finally, we demonstrate the efficacy of learning visual localization.

Datasets. Two public datasets are employed to evaluate the proposed method. We start experiments by evaluating the system on Microsoft 7Scenes dataset (117), which comprises of RGB-D images. The images were collected at resolution of 640×480 pixels from seven different indoor scenes. Multiple sequences were captured for each scene, and each sequence is captured with different camera motions. The ground truth camera poses are obtained with KinectFusion.

Another one is the Oxfor RobotCat dataset (95). It provides images collected in a large-scale outdoor environment. Additionally, this dataset reflects a large variations in weather conditions, such as the sunny and snowy days, it is challenging for some tasks based on vision.

Baselines. We compare our approach with several state-of-the-art camera localization methods.

PoseNet. Posenet uses a CNN to predict the pose of an input RGB image. The Posenet network is the GoogleNet (125) architecture with the top-most fully connected layer removed and replaced by one with a 7-dimensional output and trained to predict the pose of the image.

Scene	PoseNet (64)	LSTM-pose (135)	VidLoc (19)	MapNet (10)	Ours
Chess	0.13m, 4.48°	0.24m, 5.77°	0.18m, NA	0.08m, 3.25°	0.06m, 3.43°
Fire	0.27m, 11.30°	0.34m, 11.9°	0.26m, NA	0.27m, 11.69°	0.24m, 12.13°
Heads	0.17m, 13.00°	0.21m, 13.7°	0.14m, NA	0.18m, 13.25°	0.19m, 13.18°
Office	0.19m, 5.55°	0.30m, 8.08°	0.26m, NA	0.17m, 5.15°	0.16m, 5.14°
Pumpkin	0.26m, 4.75°	0.33m, 7.00°	0.36m, NA	0.22m, 4.02°	0.21m, 4.32°
Red Kitchen	0.23m, 5.35°	0.37m, 8.83°	0.31m, NA	0.23m, 4.93°	0.24m, 5.18°
Stairs	0.35m, 12.40°	0.40m, 13.7°	0.26m, NA	0.30m, 12.08°	0.29m, 12.36°

Table 4.1: Translation error (m) and rotation error (○) for various methods on the 7-Scenes dataset.

LSTM-pose. integrates the LSTM to explore the feature dependency.

VidlocNet. learns bidirectional mapping with shared weights in intermediate layers.

MapNet. take both the absolute pose estimation and relative pose estimation. However, it dosed not exploit the long term spatial and temporal dependency.

4.5.1 Experiments on Microsoft 7-Scenes Dataset

In this section we describe the experiments that we performed on the Microsoft 7-Scenes dataset. The results of our experiments testing the accuracy of our method are shown in Table 4.1. Following the same convention of prior work, we compute the median error for camera translation and rotation. As shown, our proposed method does improve performance. The estimated position is illustrated in Figure 4.5;

Effect of Attention. In Figure 4.3, we show some examples of images combined with the learned attention map. We can see that the discriminative regions of the images are highlighted. We observe that the discriminative regions for each images are those areas with salient objects. This suggests that our approach works as expects.

4.5.2 Experiments on RobotCar Dataset

Then we evaluate the proposed approach on the Oxford RobotCar dataset. Figure 4.6 shows the 2D multi-dimensional scaling (MDS) of the features from the second latest layer of the model trained on the LOOP sequence. The points are chronologically colored. We observed that the MDS created from the features of our model is close to be a rectangle that is consistent to the ground truth camera poses which is from a loop. This demonstrates the features extracted by our model is better correlated to the camera poses, hence improve the pose estimation accuracy.

Internal Representation. We illustrated the cumulative distributions of the translation errors on the LOOP sequence dataset in Figure 4.4. It shows that our model significantly improves the performance compared to the baseline model MapNet.

4.6 Deep Pilot

In this section, we adapt the proposed approach into a more challenging task, that is to assist in UAV flying. Traditionally, the UAV system depends on sensors, such as GPS, IMU to navigate it self and a pilot to generate control commands. Several works have been done to integrate deep neural network for UAV. (143) introduces a method to estimate horizon line in the images. (13) estimate the attitude based on horizon line. Recently, (27; 1) enhance the the Kalman Filter with deep neural network for attitude estimation. We instead take the camera as the basic data source, and predict the flight states and control commands by training a deep neural network. The overview of the proposed system is illustrated in Figure 4.7.

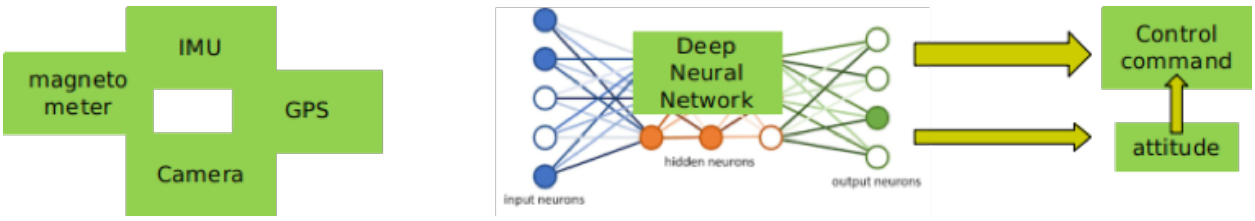


Figure 4.7: Overview of deep neural network based pilot.

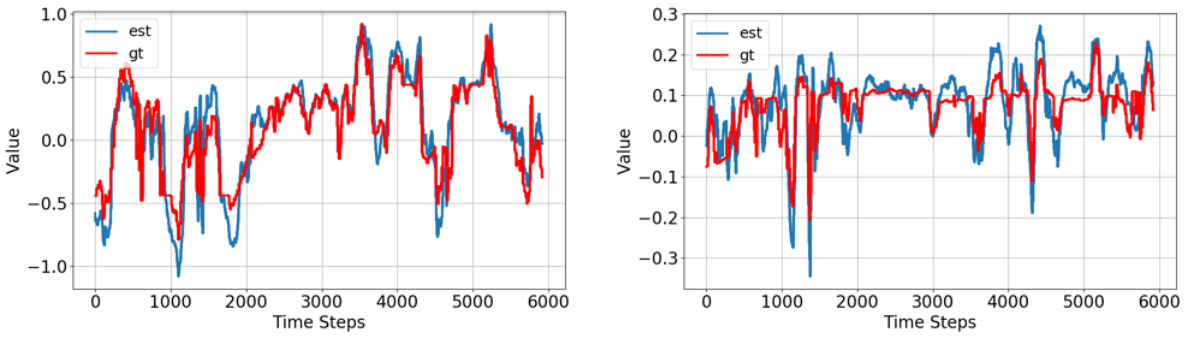
4.6.1 System Setup

The system is based on a fix-wing aircraft. The camera is mounted on the head of the aircraft. this ensures that the horizon line is recognizable in each image. We assume that the neural network is able to recognize the horizon line and takes it as the reference for attitude estimation.

Dataset. To collect data, the camera shot videos while the flight log of the pilot is recorded. We take 24 flight tests. All the flight tests are done in the same location. Each test collect 6,000 to 12,000 images. During the training stage, we feed image sequence to the model and take the flight log as the ground truth. We first align the image data with the flight log base on the time stamps. To get rid of the system uncertainty, we do not use the data during taking off and landing. Some image samples of this dataset are shown in Figure 4.8. As a preprocessing method, we normalized the values with the mean and variance calculated from the dataset.

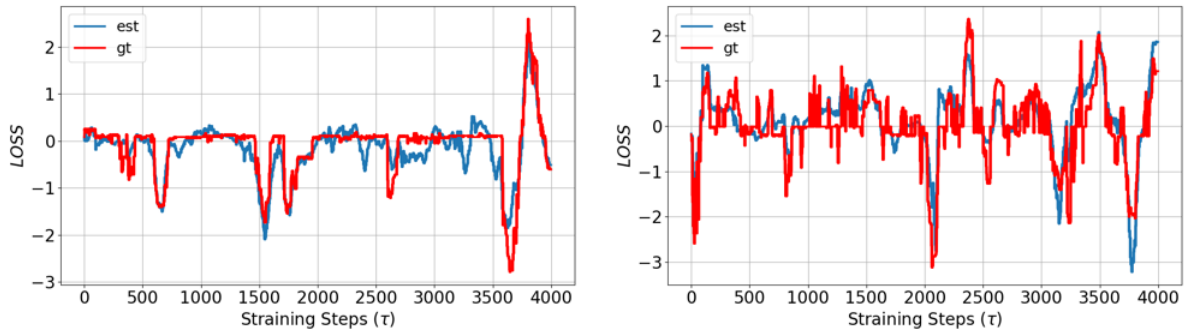
Figure 4.8: Illustration of samples from the UAV flighting dataset. The UAV flys with large variations of orientation. The horizon line is always recognizable and provides reference for UAV orientation.

Deep Pilot. Traditionally control command is created based on a well formulated dynamic model. Given the measured state and desired state, the controller is able to generate a sequence of control rule that empowers the system to follow the previously defined movement. However, it is nontrivial to exactly formulate a dynamic model. The recent deep neural network is becoming popular due to its capability of modelling nonlinear system. Without an exactly dynamic model, it is meaningful to design a deep neural network based controller in an end-to-end manner. Specifically, The deep neural network makes predictions directly on the sensor data, such as images. This model autonomously extracts feature from the inputs then estimates roll and pitch angles and their rates, finally to generate appropriate control commands to control the aircraft. The difference between the traditional pilot and the deep neural network based pilot is demonstrated in Figure 4.7.



(a) Phi. (b) Theta.

Figure 4.9: Orientation estimation results on the UAV flighting dataset. The ground truth camera trajectory is the red line, the star indicates the first frame, and the blue lines show the camera pose predictions. All the values are normalized to stabilize the training process.



(a) Elevator. (b) Rudder.

Figure 4.10: Control command estimation results on the UAV flighting dataset. The ground truth Control command is the red line, the star indicates the first frame, and the blue lines show the camera pose predictions. All the values are normalized to stabilize the training process.

Since our model is designed to regress the 6-deg pose of the UAV, we take a advantage of the proposed approach for control command prediction. Instead of using the inertial sensors to navigate the UAV and generate control command accordingly, our model takes the image and the current ϕ and θ as inputs and to predict the control command for the elevator and rudder. We modify the model slightly to estimate the control command of *elevator* and *rudder*. The result is illustrated in Figure 4.10. Specifically, we take two fully connection layers to predict them separately. And for the control command estimation, we augment the input space to be the combination of image and the current ϕ and θ.

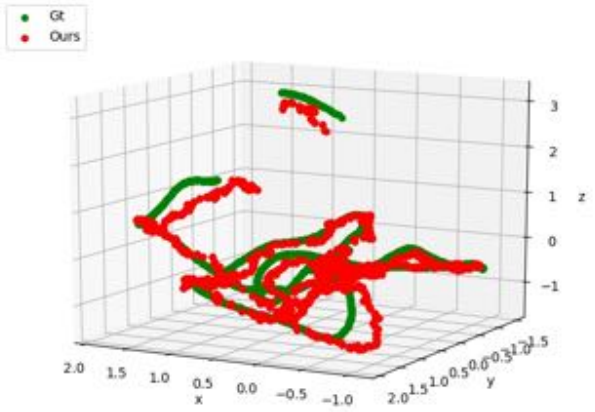


Figure 4.11: Camera localization results on the UAV flighting dataset. All the values are normalized to stabilize the training process.

4.6.2 Experiments on Flight Dataset

We then evaluate the model on the UAV flight dataset. To evaluate the model, we aim at predict the two flighting states, which are ϕ and θ. The estimated states are shown in Figure 4.9. As we can see, the predicted ϕ and θ are consistent to the ground truth. Beside, the estimated translation is shown in Figure 4.11. Both of these estimated states demonstrated that our model works well in this challenging dataset. The quantitative evaluation is listed in Table 4.2. The mean translation error is 76(m), mean orientation estimation error is 19.69° and the mean control command estimation error is 37%.

Setting	Error
Translation	76m
Orientation	19.69°
Command	37%

Table 4.2: The quantitative performance, in terms of mean estimation error, evaluated on the flighting dataset. Obviously, advantages of the method in predict the UAV flight status are revealed in this form.

4.7 Conclusion

In summary, The proposed model learns long-term temporal features for camera localization. Our models bring geometric constraints widely used in visual SLAM into DNN-based learning, which allow us to learn under the graphic constraints. Specially, we proposed a novel system for UAV navigation, which adapts the proposed method to estimate the flight states and generate control commands. We evaluate our approach on both indoor and outdoor datasets and show significantly better performance than baselines.

Chapter 5

Adaptively Denoising Proposal Collection for Weakly Supervised Object Localization

5.1 abstract

In this paper, we address the problem of weakly supervised object localization (WSL), which trains a detection network on the dataset with only image-level annotations. The proposed approach is built on the observation that the proposal set from the training dataset is a collection of background, object parts, and objects. Several strategies are taken to adaptively eliminate the noisy proposals and generate pseudo object-level annotations for the weakly labeled dataset. A multiple instance learning (MIL) algorithm enhanced by mask-out strategy is adopted to collect the class-specific object proposals, which are then utilized to adapt a pre-trained classification network to a detection network. In addition, the detection results from the detection network are re-weighted by jointly considering the detection scores and the overlap ratio of proposals in a proposal subset optimization framework. The optimal proposals work as object-level labels that enable a pseudo-strongly supervised dataset for training the detection network. Consequently, we establish a fully adaptive detection network. Extensive evaluations on the PASCAL VOC 2007 and 2012 datasets demonstrate a significant improvement compared with the state-of-the-art methods.

5.2 Introduction

Object detection, which attempts to place a tight bounding box around every object of a given image, is an important problem for image understanding. This problem has been extensively studied

in recent years (6; 7; 8; 79; 115; 122), and the state-of-the-art detection performance promotes a variety of applications, including human pose estimation (130) and crowd counting (164). One key step for object detection is to learn a distinctive representation of the objects from a large quantity of labeled data. Most existing methods rely on object-level labeled dataset (25) so that their models learn visual features from those specified regions. However, data annotation is an exhaustive and error prone work. In order to reduce the annotation cost, a common strategy is to learn the detector in a weakly supervised manner that only binary image-level labels representing the overall presence or absence of an object category are added to the images for training.

Multiple instance learning (MIL) (17; 26; 42) is an intensively used strategy in dealing with the task of weakly supervised object localization (WSL). It selects object regions of interest (proposals) from the positive images that contains the object, and learns an appearance model of the object from the features in the selected regions. This method has the tendency to get stuck in local optima. Therefore, a re-localization and re-training strategy is typically taken to push the solution close to the global optima. Pentina *et al.* (103) forms a curriculum learning strategy to feed the training process from easy images with big objects to hard images with many small objects. Shi *et al.* (115) propose a strategy that re-weights the proposals' scores according to the consistence between the proposal size and the estimated object size. Even though these strategies attempt to improve the MIL, finding positive image bags containing certain class object for MIL classifier, in some senses, depends on guessing and it is possible to take a negative bag as a positive one. It is also difficult to get tight bounding boxes exactly containing the objects. These drawbacks require strategies that adaptively refine the estimated bounding boxes to tightly contain the objects.

Another line of this research is based on convolutional neural networks (CNNs) (61; 127) that are capable of learning generic visual features generalized to many tasks. Methods in this category are inspired by the facts that, without location annotation, a pre-trained image classification CNN learns representative information of objects and object parts. Many efforts leverage CNN to extract discriminative appearance features, then train a MIL appearance model for object detection (146). Recent efforts (8; 79) achieve significant performance improvement with proposed

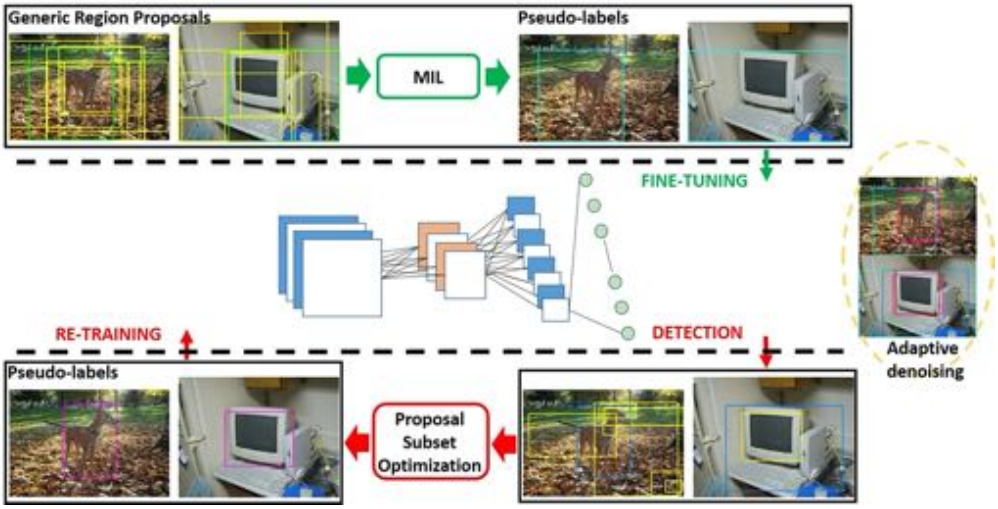


Figure 5.1: Overview of our method. We use mask-out strategy to collect the generic region proposals and take the MIL to generate pseudo labeled training set. This dataset is then fed to a WSL loop, so that the object detector is re-trained progressively. We also take the re-localization (170; 176) step by re-weighting object proposals according to the detection scores and the overlap of the proposals. Bounding boxes (in yellow) represent the confident proposals; while the bounding box in other colors in each block represents the highest confident proposal.

end-to-end methods, which adopt a pre-trained classification network to mine location information and transfer the problem from weakly supervised object localization to psudo-strongly supervised object detection. However, generating instance-level labels from the image-level labels is nontrivial since the objects from the same category may appear with different shapes and background. A pre-trained classifier makes predictions on salient features. The extracted appearance features represent object parts, which lack information on the instance as a whole. Moreover, it is different to determine the size of bounding boxes that exactly contain the objects in the feature-level searching. As a result, the obtained instance-level labels are inexact. In this paper, we propose a new framework based on two observations: (i) The proposals are a collection of background, object parts, and objects; and (ii) it is hard to train object detectors directly under weakly labeled dataset due to the substantial amount of noise in the object proposal collection and the size variation of the objects. Our method integrates several strategies to adaptively eliminate the noise in the object proposal collection. We take an enhanced MIL algorithm, which is proceeded by a mask-out strategy to mine the proposal collection and fine-tune a pre-trained classification network

through re-weighting and re-training, which exploits proposal subset optimization (165) to further re-weight the detection results.

Our re-weighting and re-training strategy aims at determining the optimal proposals automatically. To this end, we take a subset optimization method to select object proposals. It is based on both the detection scores from the pre-trained detection network and the overlaps between the candidate bounding boxes. This strategy puts higher weights on proposals that have large overlap area with others. Specially, We reweight those object proposals with high detection scores according to how much the bounding box overlaps with other bounding boxes. Iteratively, we utilize this subset optimization method to improve the re-localization step.

This re-weighting scheme reduces the uncertainty in the proposal distribution, making the re-weighting step more likely to pick a proposal correctly covering the object. Fig. 5.2 shows an example of how the subset optimization changes the proposal score induced by the current object detector, leading to a more accurate localization.

Our contributions are as follows: (i) We propose a novel work flow to collect confident proposals, which integrates the mask-out strategy, MIL, and subset proposal optimization. The MIL model is trained on the selected proposals of mask-out strategy and mines confident proposals to reduce the background clutters and potential confusion from similar objects cases. The subset proposal optimization further refines the proposals by re-scoring the bounding box; (ii) Following the idea of re-localization and re-training, the candidate proposals are refined based on both the detection scores and the overlap ratios between the proposals. We then iteratively adapt a pre-trained classification network to a detection network with those quality enhanced proposals. This is a new pipeline for improving object proposals; And (iii) detailed evaluations on the PASCAL VOC 2007 and 2012 datasets (34) demonstrate that our weakly supervised object detection with adaptively denoised proposal collection performs favorably against the state-of-the-art methods. The proposed model and trained parameters will be available on the authors website.

5.3 Related work

Extracting meaningful information from the environment is a challenging task (150; 149; 155). In recent years, deep neural networks are becoming more and more popular for knowledge discovering in many computer vision tasks, such as object detection (137; 88), visual question answering (161), pose estimateion (53; 52; 160), image synthesis (152; 151; 153), face recognition (14), and depth estimation (46; 47). Object detection is the task of recognizing and localizing the objects in the images with the deep model trained on labelled ground truth (94). However, labelling the images with bounding box for each object is a nontrivial work. In the scenario of weakly supervised localization, the training images are known to containing instances of a certain object class but their locations. There is no ground truth bounding box available for each object in the training dataset. The task is both to localize the objects (estimate the bounding boxes tightly containing the instances) and to classify the objects. What we have are the image-level annotations which are weak supervision for localizing the objects. To train a detection network with image-level supervision, we need first to localize objects in all the images of the training dataset based on image-level annotations, and then use the localization results to train a detector for the test set. The WSL problem is often handled with multiple instance learning (MIL) (7; 8; 18; 42; 116; 122), where the images are treated as bags of object proposals (132; 177) (which are bounding boxes estimated to localize the objects). A negative image dose not contain instances of certain category. A positive image contains at least one positive instance, mixed in with a majority of negative ones. The goal is to find the true positive instances from which to learn a classifier for proposal classification.

Previous works achieve significant improvement by exploring ways to enhance the MIL. Siva *et al.* (120) propose an effective negative mining approach combined with discriminative saliency measures. Song *et al.* (122) formulate an initialization strategy for WSL as a discriminative submodular cover problem in a graph-based framework, and develop a negative mining technique to increase robustness against incorrectly localized boxes (123). Bilen *et al.* (7; 8) propose a relaxed version of MIL that softly labels object instances instead of choosing the highest scoring ones. They also propose a discriminative convex clustering algorithm to jointly learn a discriminative

Figure 5.2: Detection results from NMS (red line in left) and subset optimization (center). Bounding boxes (BB) (right) represent the highest confident proposals got from different steps (blue BB: CNN, green BB: maskout, pink BB: re-train, cyan BB: MIL). We compare the detection results by bounding boxes in different colors, which shows our re-training strategy is able to get the denoising proposals by re-weighting object proposals according to the detection scores and the overlap ratios of the proposals.

object model and enforce the similarity of the localized object regions.

As CNNs have turned out to be surprisingly effective in many vision tasks including classification and detection, recent state-of-the-art WSL approaches also build on CNN architectures (8; 173) or CNN features (18). Bilen *et al.* (8) modify a region-based CNN architecture (40) and propose a CNN with two streams, one focuses on recognition and the other one on localization, which performs simultaneously region selection and classification. Similarly, Li *et al.* (79) use the MIL to obtain the initial detection results and propose a domain adaption method (51; 102) to fine-tune a classification network into a detection network with the initial detection results. The results show a performance improvement on the detection accuracy. Shi *et al.* (115) attempt to score the proposals by the size and retrain the detection network with the re-weighted proposals according to an easy to hard order, based on the assumption that the proposals with bigger size provide more information to train the network than the those with smaller size. Our work is related to these CNN-based MIL approaches that perform WSL by end-to-end training from image-level labels. In contrast to the above methods, however, we focus on a CNN architecture that is re-trained in an order for detection accuracy improvement with denoised proposals.

The concept of adaptive learning in an order was also studied in computer vision (72; 103; 112; 115; 131). These works focus on a key question: how to re-weight the proposals? Sharmanska *et al.* (112) employ some privileged information to distinguish between easy and hard examples in

an image classification task. The privileged information are additional cues available at training time, but not at test time. They employ several additional cues, such as object bounding boxes (44; 106), image tags and rationales to define their concept of easiness (72). Lai *et al.* (71) select highly confident object proposals under the guidance of class-specific saliency maps. Pentina *et al.* (103) consider learning the visual attributes of objects. Shi *et al.* (115) propose a size estimator to re-weight the proposals based on the size of the instances in the image. They use curriculum learning in a WSL setting and propose object size as an "easiness" measure. Shi *et al.* (114) consider the task of discovering object classes in an unordered image collection. their model is initialized with regions of "stuff" categories, and is then used to support discovering "thing" categories in unlabelled images with the help of a fully supervised segmentation model. Bodla *et al.* (9) propose a soft method to select the bounding boxes. They decay the classification score of a box which has a high overlap with top-scoring boxes, rather than suppressing it. Jie *et al.* (61) explore the Fast RCNN model (40) and propose a self-taught learning method for proposal selection. The most related work to ours is the very recent study (127), which designs an on-line classifier refinement pipeline to progressively locate the most discriminative region of an image. By contrast, we propose a novel work flow to adaptively refine the proposals, i.e., to iteratively collect a more confident subset of proposals. In addition, we take the re-training strategy to fine-tune the model with the denoised proposal subset. The proposed work flow, by integrating several novel proposal mining strategies, makes it adaptable to a variety of weakly supervised object detection tasks.

5.4 Adaptively Denoised Proposal Collection

The proposed weakly supervised object detection method is illustrated in Fig. 5.1. This model consists of three major components, namely confident proposal learning, object detector learning and proposal subset optimization. They are successively employed to adaptively refine the proposal collection. The remainder of this section discusses these three components in details.

5.4.1 Confident Proposal Mining

We consider the weakly supervised object localization problem as an adaptively proposal denoising procedure that gradually refines the proposal collection. At the end, we transfer the problem from the weakly supervised object localization to a pseudo-strongly supervised object detection. Based on a pre-trained CNN classification network and a MIL model, our work flow adaptively selects confident proposals other than those comprised of background or object parts from the candidate proposals generated by EdgeBoxes (177).

Assisted by the classification network, we first utilize the mask-out strategy to collect object proposals. The idea of masking out the input of CNN has been previously explored in (163), which replaces the pixel values of the proposals with fixed mean pixel values; and compares the classification scores of feeding the real image and its mask-out images into the classification network. Intuitively, if the mask-out image introduces a notable drop in the classification score for the c_{th} class, the region can be considered as containing an object belonging to the c_{th} class. Inspired by (79; 163), we apply the mask-out strategy to select the proposals containing a certain object. We denote the classification network as f_c that maps an image to a confidence vector of c_{th} classes. The confident proposals B_c are selected by investigating the difference of classification score between the selected image $I(x)$ and its mask-out image $I(x/b)$. This is formulated as

$$B_c = arg\max_b(f_c(I(x)) - f_c(I(x/b))) \tag{5.1}$$

where b represents the masked-out region. To select confident proposals, we first set a threshold on the classification score. The region b is considered discriminative for c_{th} class based on two aspects: the score of classifying the image $I(x)$ to the c_{th} class is beyond the threshold and the classification score drop between the image and corresponding mask-out images is maximum.

Once the proposals are obtained by applying the mask-out strategy, we separately learn one MIL model for each category. Taking the purified proposals selected by the mast-out strategy as training dataset makes the basic MIL initialized from a higher baseline, which not only stabilizes

the training process, but also reduces the time for training (79). In the MIL model, each instance is described by a feature vector. More specifically, each feature vector is regarded as an instance and each image is represented by a bag of instances. For instance, the training image x_i is considered as a bag of proposals with pseudo strong labels $y_i \in \{-1, 1\}$ indicating whether the bag contains an instance in the specific category. A bag is considered to be negative if there is no instances or all its instances are not in that category, while it is positive if there is at least one of its instances in that category. Given feature representation $\phi(x_i, z)$, we iteratively train the MIL model with the objective written as

$$\min_{w \in R} \frac{1}{2}||w||_2^2 - \sum_{i=1}^{n} log((y_i \max_{z \in \mathscr{Z}} w^T \phi(x_i, z) - \frac{1}{2}) + \frac{1}{2}) \tag{5.2}$$

where w represents the parameters of the MIL model and z is called the "latent variable" chosen from the set Z, which is typically a set of bounding boxes. The top-scoring proposals given by the mask-out strategy are taken as positive samples for each category, which are used to train the MIL model. Among the initial bounding boxes, the set Z contains all possible candidate instances. Maximizing the objective function over Z amounts to choosing a bounding box containing the whole object. The proposal, in this work, is represented by a 4096-dimensional feature vector from the second-last layer of the classification network.

The top row in Fig.5.1 demonstrates the idea of the confident proposal mining, which starts from the mask-out strategy and ends with the high confident output from MIL.

5.4.2 Proposal Subset Optimization

Proposal selection based object detection method has one severely issue of overlapping among the bounding boxes that correspond to the same object. To select the best bounding box for each object, greedy non-maximum suppression (NMS) is widely employed as the latest strategy which selects the top-scoring bounding box b_i and discards other bounding boxes $\mathscr{M}$ that have overlaps with the chosen one larger than a threshold T. Due to simplicity, this NMS mainly focus on the

detection score s_i. By taking the Intersection over Union (IoU) as the measure of overlapping, this non-maximum suppression process can be described as

$$s_i = \begin{cases} s_i & IoU(\mathscr{M}, b_i) < T; \\ 0 & IoU(\mathscr{M}, b_i) \geq T. \end{cases} \tag{5.3}$$

However, there are no instance-level labels available for network training in the weakly supervised localization task, even the bounding boxes estimated with top score are tended to be noisy. To overcome this issue, we propose a subset optimization scheme. It is realized by re-weighting the detection scores among the bounding boxes with high but noisy initial scores, where greedy NMS is not able to adjust the estimated bounding box accordingly. The proposed approach is similar to that described in (165). However, we employ the method to solve the weakly supervised learning problem. The confident proposals with high detection scores are grouped into clusters by jointly considering the scores and the spatial overlaps between the proposals. The bounding box set is represented by $B = \{b_i : i = 1 : n\}$. We denote the group membership as $X = (x_i)_{i=1}^n$, where $x_i = j$ if b_i belongs to a cluster b_j. Then one exemplary bounding box o is selected from each cluster B as the final output. This is formulated as finding the maximum a posterior (MAP) solution of the joint distribution $P(O, X|I; B, S)$, which tends to assign big value to the bounding boxes that have large overlap with more confident proposals. After taking the log of the posterior, the objective function becomes:

$$O = X^* = arg\max_X \sum_{i=1}^{n} \omega_i(x_i) \tag{5.4}$$

where $\omega_i(x_i = j) = logP(x_i = j|I)$

$$\mathrm{P}(\mathrm{x_i} = \mathrm{j}|\mathrm{I}) = \begin{cases} Z_2^i \lambda & if\ j = 0; \\ Z_2^i K(b_i, b_j) s_i & otherwise. \end{cases} \tag{5.5}$$

$K(b_i, b_j)$ is the window IoU used to measure the spatial overlap between b_i and b_j, $S = \{s_i : i = 1 : n\}$ is the score set containing the detection scores of all the bounding boxes, and Z_2^i is the normalization constant. Parameter β and γ control the penalty level. Note that our proposal subset optimization method takes both the scores and the overlapping into consideration since the detection scores in the weakly supervised task are not always reliable.

The proposal subset optimization problem is defined as:

$$O^* = arg\max_{O} \beta \sum_{i \in \hat{O}} s_i - \gamma \sum_{i,j \in \hat{O}: i \neq j} K(b_i, b_j) \tag{5.6}$$

In this setting, we first maximize the objective function over X according to Eq. (5.4), which will select the cluster centers. Then, a greedy algorithm is used to choose a minimal number of bounding boxes as the outputs based on Eq. (5.6). More details of the method can be found in (165).

5.4.3 Object Detector Learning

In this step, we adapt the pre-trained classification network to an object detection network. This neural network is trained with the pseudo labeled proposals obtained from the proposal subset optimization strategy. We employ the re-weighting and re-training strategy for network adaption. The network parameters are fine-tuned for object localization, as illustrated in the bottom of the Fig.5.1. We organize it as adaptively refining the proposal subset, which is similar to the curriculum learning. However, we do not separate the training dataset into easy and hard parts. We start by running MIL, which is initialized with the results from mask-out strategy. This leads to a reasonable first detection model A_1. We move forward by running proposal subset optimization

on the proposals subset with high detection scores, which produce a re-weighted proposal subset. The process then moves on to the second training iteration, where the training dataset consists of re-weighted proposals with more confident pseudo labels. As a result, the refined model A_2 will localize the objects better than A_1, as it is trained with better supervision in the re-training step. The process iteratively moves on to the next round, starting from the detection model A_k and yielding a better one A_{k+1}. The whole training procedure is described in Algorithm 1.

The selected results from each strategy are shown in Fig. 5.2. It is demonstrated that the bounding boxes selected by the fully adapted detection network exactly contain the objects, while the bounding boxes selected by mask-out strategy and MIL contain the object but with a large margin. By re-weighting the confident proposals according to the detection scores and the overlap of the proposals, the re-training strategy is able to generate more confident proposals.

Algorithm 1 The training pipeline of the proposed algorithm.

Input:

Images $x \in X$; $B = \{b_i : i = 1 : n\}$ candidate boxes; $S = \{s_i : i = 1 : n\}$ the corresponding scores; K, the refinement times; M, the network iteration times; θ_E, network parameters.

Output:

A fully adaptive detection network.

1: Classify the real images and the mask-out images with the classification network; select the top M proposals by Eq. (5.1) as the initial proposal set $P_0 \leftarrow \{x, s_0, B_0\}$.
2: For each category, construct positive and negative bags within S_0; train the MIL model and collect the detection results from the trained MIL model as proposal set $P_1 \leftarrow \{x, s_1, B_1\}$.
3: **for** $k = 1$ **to** $K - 1$ **do**
4: Set $P \leftarrow P_k$.
5: **for** $m = 1$ **to** $M - 1$ **do**
6: Sample $P \rightarrow \{x, s, b\}$ as a minibatch.
7: Network forward propagation and get loss ℓ.
8: Network backward propagation, $\theta_E \overset{+}{\leftarrow} -\nabla_{\theta_E}(\ell)$.
9: **end**
10: Collect the detection results from the trained detection network, $P'_k \leftarrow \{x_k, s_k, B_k\}$.
11: Choose the proposals with subset optimization by Eq. (5.6); update the proposal set $P_{k+1} \leftarrow \{x, s'_k, B'_k\}$.
12: **end**

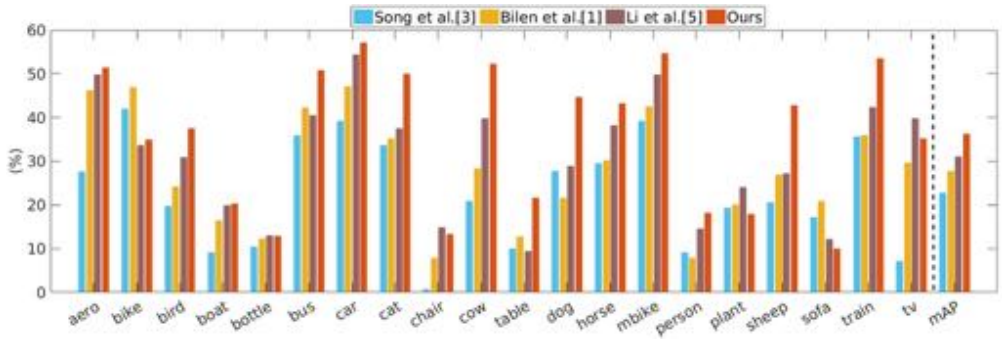


Figure 5.3: A comparison of our method (AlexNet) of detection mean average precision (mAP) on the PASCAL VOC 2007 dataset. Our method with the mAP (36.1%) significantly outperforms other methods for most of the categories.

5.5 Experimental Evaluation

Dataset and settings: The proposed approach is extensively evaluated on two publicly available datasets: PASCAL VOC 2007 and 2012 datasets. Both of them have 20 classes of different images. We employ both the AlexNet (68) and VGGNet (119) as our base CNN models, initialized with parameters transferred from the classification network, which is pre-trained on the ImageNet dataset. As an initialization step for class-specific proposal mining, we use Edge Boxes (177) to generate 2,000 object proposals for each image. The mask-out strategy is first utilized to remove most of the noisy proposals and return top 50 confident proposals. These selected proposals work as the input for multiple instance learning. At the re-training stage, network is trained by employing the SGD solver with the learning rate of 0.0001 for 40k iterations.

Evaluation metrics: To quantitatively evaluate the performance of the proposed method, we take two types of metrics, which are applied at the training and testing stage respectively. In the training dataset, we compute the percentage of images from which we obtain correct localization (CorLoc) (26). In the test dataset, we evaluate the performance of the object detector using mean average precision (mAP), a standard metric used in PASCAL VOC. Within both the metrics, we consider that a bounding box is correct if it has an IoU ratio of at least 50% with the ground-truth object annotation.

Comparison with the state-of-the-art algorithms: We compared the proposed algorithm

Methods (VOC 2007)	CorLoc	mAP
Li et al. (79)	49.8	31.0
Shi et al. (115) (AlexNet)	**60.9**	<u>36.0</u>
Our scheme	<u>53.4</u>	**37.2**
Li et al. (79)	52.4	39.5
Shi et al. (115)	**64.7**	37.2
Bilen et al. (8) (VGGNet)	53.5	34.8
Jie et al. (61)	56.1	40.8
Tang et al. (127)	<u>60.6</u>	**41.2**
Our scheme	55.9	<u>40.9</u>
Methods (VOC 2012)	CorLoc	mAP
Li et al. (79)	-	22.4
Our scheme (AlexNet)	-	**25.3**
Li et al. (79)	-	29.1
Jie et al. (61)	54.8	**38.5**
Tang et al. (127) (VGGNet)	**62.1**	<u>37.9</u>
Our scheme	<u>55.2</u>	35.2

Table 5.1: Quantitative comparison in terms of detection mean average precision (mAP) on the PASCAL VOC 2007 test set and correct localization (CorLoc) on the PASCAL VOC 2007 trainval set using AlexNet or VGGNet. The last rows show the mAP on the PASCAL VOC 2012 val set. We highlight the best performances and underline the 2nd best performances.

with the state-of-the-art methods dealing with the weakly supervised object localization problem (7; 79; 115; 122). None of them use strong labels for training.

Fig. 5.3 shows the performance comparison between our proposed method developed with the AlexNet as baseline and the state-of-the-art WSL works (7; 79; 122) on the VOC 2007 dataset. Models from Song *et al.* (122) and Bilen *et al.* (7) are MIL-based approaches with advanced model initialization. Our method is developed based on that from Li *et al.* (79). Moreover, Tang *et al.* (127) propose an on-line instance classifier refinement, which classifies a fixed-size conv feature produced by some convolutional (conv) layers with spatial pyramid pooling (SPP) layer. As the classifier is trained with the features from the SPP net, this model takes the advantage of a better initialization. In an entirely different way, we progressively adapt a classification network to an object detection network with denoised proposals as the pseudo strong labels. Such domain adaptation helps to learn a better object detector from image-level annotated data. Unlike previous

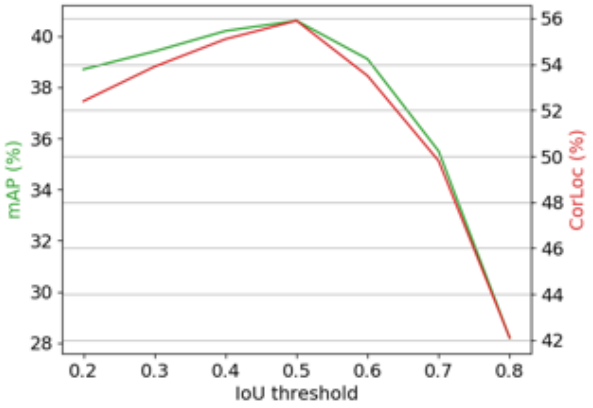


Figure 5.4: Performance over different IoU threshold of the VGG16 version on PASCAL VOC 2007

Re-train (mAP)	0_{th}	1_{st}	2_{nd}	3_{rd}	4_{rd}
AlexNet	31.0	34.1	36.8	37.2	37.2
VGGNet	38.5	39.9	40.3	40.9	40.9

Table 5.2: Quantitative comparison in terms of detection mean average precision (mAP) on the PASCAL VOC 2007 test set for different re-training steps with AlexNet or VGGNet.

works that rely on noisy proposals to localize the object candidates, we mine finer and class-specific proposals from the proposed work flow, which integrates the mask-out, MIL and subset proposal optimization. In addition, a fully model adaption is guaranteed with the re-training and re-weighting strategy.

By incorporating the proposal subset optimization, the proposed model significantly outperforms other methods in terms of mAP for most of the categories. In Table 5.1, we make comparisons in terms of both the CorLoc and mAP on the training and testing set of the VOC 2007 dataset, respectively. In addition, we present the mAP on the val set of the VOC 2012. For other baseline methods, we list the best performances of the AlexNet and VGGNet models, which are reported in the paper. Based on the VGGNet, our method achieves 40.9% mAP on VOC 2007 test set and 35.2% mAP on VOC 2012 val set. It is also evident from Table 5.1 that the detection performance is significantly improved by using a deeper network. Note that the method introduced by jie *et al.* (61) is a regional CNN detector (Fast R-CNN (40)). This model trained on seed samples is suf-

Strategy	BBox-initialization		Re-weighting	Re-training		
	Mask-out	MIL	Subset Optimization	AlexNet	VGG16	VGG19
VOC 2007	3	24	2	4	7	9
VOC 2012	7	36	3	7	14	17

Table 5.3: Quantitative comparison in terms of computational time (hour) on the PASCAL VOC 2007 and 2012 training sets for different strategies.

ficiently powerful for selecting the most confident tight positives and is able to further train itself with the optimized proposals. We compare our method against this Fast RCNN based method by listing the results in Table 5.1. A similar performance is obtained by our model as the one on VOC 2007.

In addition to the standard IoU for evaluation, we analyze the influence over different IoU threshold in Fig 5.4. It is evident that setting IoU = 0.5 achieves the best performance, and the results are not very sensitive to different values: when changing it from 0.5 to 0.6, the performance only drops a little bit.

Impact of re-training strategy: The re-training strategy we utilized so far is straightforward. The process is to establish an order that adaptively optimize the refined proposals, and then fine-tune the detection network with the confident proposals. We notice that the proposals used to fine-tune the network are critical to train the baseline for detection. So it is promising to improve the annotation through an adaptive way.

We use the same settings during the re-training stage as we adapt the classification network to a detection network. After training the detection network, we select the top 30 detection results and optimize them with the proposal subset optimization. Consequently, the training dataset is adaptively denoised and we obtain a better detection network. Table 5.2 demonstrates that the mAP is increased from 31.0% to 37.2% for the AlexNet and from 38.5% to 40.9% for the VGGNet.

Computational time analysis: We report the evaluation results on PASCAL VAL 2007 and PASCAL VAL 2012 in the paper. The re-training is conducted under AlexNet, VGG16, and VGG19. The training time of the experiments largely depends on the hardware resource. We train and evaluate the proposed method using the Intel Xeon(R) CPU E5-1607 v2 @ 3.00GHz $\times$

Figure 5.5: Sample detection results. Green boxes indicate ground-truth annotation. Red boxes indicate correct detections (with IoU ≥ 0.5). The sample images show the correct detections from different classes.

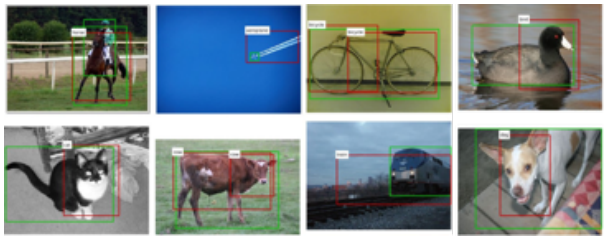


Figure 5.6: The sample images shows the wrong detections due to imprecise detection. Green boxes indicate ground-truth annotation. Red boxes indicate imprecise detections (with IoU < 0.5).

4 and four K80 GPUs with 12 GB memory on a cluster. To reduce the training time of MIL, we employ 12 CPUs to separately train the MIL for each category of 21 classes. The training time of the experiment is shown in Table 5.3.

Error analysis: Fig. 5.5 shows some samples with accurate detections and Fig. 5.6 shows several examples with wrong detections. Our model often detects the correct objects in the image since we train the detector by incorporating a proposal subset optimization to improve the inaccuracy of the localization. Most of the model for WSL task may fail to predict a sufficient tight bounding box (79). The adaptive denoising part of Fig. 5.1 illustrates the procedure that the proposals are adaptively selected so that they gradually converge to the ground-truth of annotations. Nonetheless, the proposed model still has limitation as shown in the wrong detections in Fig. 5.6. This is because our proposal subset optimization also depends on the detection scores even though it incorporates the overlaps of the proposals.

5.6 Conclusion

We have proposed a novel model by integrating adaptive proposal denoising strategies to handle the weakly supervised object localization problem. This approach first selects confident proposals by utilizing the output of the MIL framework as the starting point of training the detection network. At the training stage, we first adapt a pre-trained classification network with high confident proposals to a detection network, then re-weight the detection results with the proposal subset optimization method. The re-weighted proposals are taken to re-train the network, resulting a detection network that achieves competitive performance on PASCAL VOC datasets. As a follow-up study, it is desire to adapt a new feature extraction method for the weakly supervised localization task. It is interesting to add the attention mechanism that assists to obtain attended features. We would like to introduce a module that effectively and efficiently extracts purified features.

Chapter 6

Towards Learning Affine-Invariant Representations via Data-Efficient CNNs

6.1 abstract

In this paper we propose integrating a priori knowledge into both design and training of convolutional neural networks (CNNs) to learn object representations that are invariant to affine transformations (*i.e.* translation, scale, rotation). Accordingly we propose a novel multi-scale maxout CNN and train it end-to-end with a novel rotation-invariant regularizer. This regularizer aims to enforce the weights in each 2D spatial filter to approximate circular patterns. In this way, we manage to handle affine transformations in training using convolution, multi-scale maxout, and circular filters. Empirically we demonstrate that such knowledge can significantly improve the data-efficiency as well as generalization and robustness of learned models. For instance, on the Traffic Sign data set and trained with only 10 images per class, our method can achieve 84.15% that outperforms the state-of-the-art by 29.80% in terms of test accuracy.

6.2 Introduction

Recently Sabour *et al.* (107) proposed a new network architecture, CapsNet, and a dynamic routing training algorithm to connect the capsules (50), a new type of neurons that output vectors rather than scalars in conventional neurons, in two adjacent layers and group similar features in higher layers. Later on Hinton *et al.* (49) proposed another EM-based routing-by-agreement algorithm for training CapsNet. In contrast to CNNs, the intuition behind CapsNet is to achieve ”viewpoint

invariance" in recognizing objects for better generalization which is inspired by inverse graphics (Hinton). Technically, CapsNet not only predicts classes but also encodes extra information such as geometry of objects, leading to richer representation. For instance, in (49), 4×4 pose matrices are estimated to capture the spatial relations between the detected parts and a whole. Unlike CNNs the performance of CapsNet on real and more complex data has not been verified yet, partially due to the high computation that prevents it from being applicable widely.

In fact exploring such invariant representations for object recognition has a long history in the literature of both neural science and computer vision. For instance, in (58) Isik *et al.* observed that object recognition in the human visual system is developed in stages with invariance to smaller transformations arising before invariance to larger transformations, which supports the design of feed-forward hierarchical models of invariant object recognition. In computer vision part-based representation (*e.g.* (36)) is one of the most popular invariant object representations that considers an object as a graph where each node represents an object part and each edge represents the (spatial) relation between the parts. Conceptually part-based representation is view-invariant in 3D and *affine-invariant* (*i.e.* invariant to translation, scale, and rotation) in 2D. Although the complexity of part-based models in inference on general graphs could be very high (22), for tree structures such as star graphs this complexity can be linear to the number of parts (35). Girshick *et al.* (41) has shown that such star graph part-based models can be interpreted as CNNs.

In this paper we aim to study the following problem: *Can we design and train CNNs to learn affine-invariant representations efficiently, effectively, and robustly?*

Motivation. Besides CapsNet, we are also partially motivated by the works such as (3; 171) that utilize *a priori* knowledge as guidance to design and train neural networks efficiently and effectively. For instance, (3) proposed the notion of "neural module" to conduct certain semantic functionality using deep learning for visual question answering. Such modules can be reusable to comprise complex networks to perform certain tasks. The semantics and the network design here come from the compositional linguistic structure of questions. Thanks to these modules, the network design is much more understandable by checking whether the outputs of a module

follow what we expect. (171) proposed encoding network weights as well as the architecture into a Tikhonov regularizer by lifting the ReLU activations, and accordingly developed a block coordinate descent algorithm for fast training of deep models.

In contrast to *a posteriori* knowledge such as visualization of learned filters in (166), a priori knowledge based approaches are more likely to be model-driven so that one can derive by reasoning alone, rather than being data-driven, in terms of building automatic systems such as neural networks. In this way, the networks with a priori knowledge are expected to be much easier to be understood by human, and their performance is more predictable and robust.

Contributions. Thanks to the convolution, CNNs are translation equivariant. This capability has contributed significantly to their widespread success. They, however, are not efficient or effective to capture the scaled or rotated objects, and thus enhancing CNNs with the capability of learning scale-invariant and rotation-invariant features is very challenging but appealing.

In this paper we design a novel deep multi-scale maxout (43) CNN to learn scale-invariant representations. We then propose training this network end-to-end with a novel rotation-invariant regularizer. To our best knowledge, we are the first to propose such regularization for handling rotation in deep learning. Note that we take the multi-scale maxout block and the regularizer as a priori knowledge for learning affine-invariant representations. Empirically we demonstrate the benefit of integrating such knowledge with network design and training, leading to better generalization, data-efficiency, and robustness of deep models than the state-of-the-art in learning affine-invariant representations.

6.3 Related Work

Scale-Invariant Networks. One simple way to handle the scale issue is using image pyramid in deep learning (85). Some works (156; 62; 126; 154) are particularly interested in extracting scale-invariant features from the networks. More broadly, multi-scale convolutional filters (or multi-kernels) are employed in networks (84; 138; 4; 75). The inception module in GoogLeNet (125)

is able to capture multi-scale information with maxout units. A similar idea has been explored in TI-Pooling (73). ResNet (45) manages to capture multi-scale information using skip connection. Multi-scale DenseNet (55) proposes using a two-dimensional multi-scale convolutional network architecture that maintains coarse-level and fine-level features throughout the network. Note that with the increase of the number of hidden layers all the CNNs tend to extract deep features within multiple scales to a certain degree.

Rotation-Invariant Networks. Recently quite a few works focus on learning rotation-invariant features using deep networks. Cohen and Welling (20) proposed Group equivariant CNNs (GCNN) by exploiting larger groups of symmetries, including rotations and reflections, in the convolutional layers. Worrall *et al.* (144) proposed Harmonic Networks by replacing regular CNN filters with circular harmonics and returning a maximal response and orientation for every receptive field patch. Both works argue that rotating the data point is equivalent to rotating the filters. Therefore, they manage to learn rotation-invariant filters in a continuous space. In contrast, some other works such as (175; 169; 91; 54; 168; 96; 140) propose learning the filters in a discretized space by quantizing the rotation angles with predefined numbers (*e.g.* from 0 to 2π, step by $\frac{\pi}{4}$) so that the final features encode the rotation information. For instance, Rotation Equivariant Vector Field Networks (RotEqNet) (96) was proposed by applying each convolutional filter at multiple orientations and returning a vector field that represents magnitude and angle of the highest scoring orientation at every spatial location.

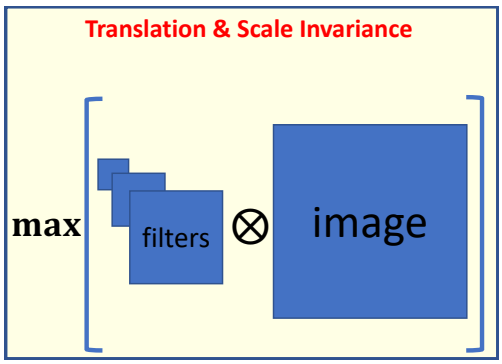
Interpretable Networks with A Priori Knowledge. Andreas *et al.* (3) proposed neural modules to mimic some basic semantic functionality using deep neural networks, based on which larger networks are constructed for specific tasks using the knowledge from natural language processing (NLP) such as grammar graphs as guidance. Belbute-Peres *et al.* (24) proposed embedding structured physics knowledge into larger systems as a differentiable physics engine that can be integrated as module in deep neural networks for end-to-end learning. Amos *et al.* (2) proposed using Model Predictive Control (MPC) as a differentiable policy class for reinforcement learning in continuous state and action spaces that leverages and combines the advantages of model-free

and model-based approaches. They also showed that their MPC policies are significantly more data-efficient than a generic neural network.

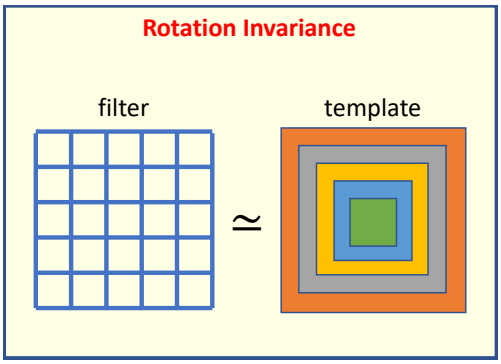
Other Related Networks. Dilated convolution (159) supports exponential expansion of the receptive field (*i.e.* window) without loss of resolution or coverage and thus can help networks capture multi-scale information. Deformable Convolutional Networks (DCN) (23) proposed a more flexible convolutional operator that introduces pixel-level deformation, estimated by another network, into 2D convolution. Spatial Transformer Networks (STN) (59) learn affine-invariant representations by sequential applications of a localization network, a parameterized grid generator and a sampler. Dynamic Filter Networks (DFN) (60; 145) was proposed to learn to generate (local) filters dynamically conditioned on an input that potentially can be affine-invariant.

Data Augmentation. It is a well-known technique in deep learning for reducing the filter bias during learning by generating more (fake) data samples based on some predefined rules (or transformations) such as translation, scaling, rotation and random cropping. Trained with such augmented data, one can expect that the networks may be more robust to the transformations. For instance, TI-Pooling (73) assembles all the transformed instances from the same data point in a pool and takes the maximal response for classification. STN (59) learns to predict a transformation matrix for each observation that can be used to augment data.

Loss Functions. From the perspective of the feature space, affine-invariant representations for an object under different transformations with translation, scale, and rotation should be mapped into a single point in the feature space ideally, or a compact cluster. To achieve this, several loss functions were proposed. For instance, the center loss (142) enforces the features from the same class to be close to the corresponding cluster center. Similar ideas have been explored in few-shot learning with neural networks (121) as well. In fact well-designed networks can generate compactly clustered features for each class with good discrimination, even if trained without such specific losses. Also such losses do not aim to learn affine-invariant features, explicitly or implicitly. Empirically we do not observe any improvement using the center loss over the cross-entropy loss, and thus we do not report the performance using the center loss.



(a) Multi-scale maxout block



(b) Regularization

Figure 6.1: To learn affine-invariant representations, we propose **(a)** a multi-scale maxout convolutional network block to handle translation and scale, and **(b)** a regularizer to handle rotation. We use (a) for constructing our network, and embed (b) into our learning.

In contrast to these previous works, we handle scale and rotation jointly in CNNs for learning affine-invariant representations. We introduce a priori knowledge into network design and training as interpretability in deep models. We demonstrate better generalization, data-efficiency, and robustness of our approach than the state-of-the-art networks.

6.4 Our Approach

Overview. To achieve translation and scale invariance, we propose a multi-scale maxout block as shown in Fig. 6.1(a), a set of filters with different predefined sizes are applied to images with convolution, and then the maxout operator is used to locate the maximum response per pixel among the filters. Mathematically this block can be formulated as

$$\max_{\omega \in \Omega} \left\{ \omega \otimes \mathbf{I}_{ij} \right\}, \forall (i,j), \tag{6.1}$$

where $\otimes$ denotes the convolution operator, $\omega \in \Omega$ denotes a 2D spatial filter, $\mathbf{I}$ denotes an image, and $\omega \otimes \mathbf{I}_{ij}$ denotes the scalar output of the convolution at pixel (i,j).

In contrast to rotation-invariant networks such as RotEqNet, there is no rotation constraint on the design of network architectures including filters. Instead, we impose such constraint on learning with our rotation-invariant regularizer. Similar to other regularizers, ours encodes the

prior knowledge of filters that we would like to learn (denoted as the template in Fig. 6.1(b)). Inspired by Harmonic Networks, ideally the learned filters should be symmetric along all possible directions, like circles. Due to the discretization of images, however, we propose an alternative to represent such symmetry that can be learned efficiently and effectively.

Learning Problem. In this paper we consider the following optimization problem:

$$\min_{\omega\in\Omega,\theta\in\Theta}\sum_{i}\ell\Big(y_i,\phi(x_i,\omega)\Big)+\lambda_1\mathscr{R}_1(\omega)+\lambda_2\mathscr{R}_2(\omega,\theta), \tag{6.2}$$

where $\{x_i,y_i\}\subseteq\mathscr{X}\times\mathscr{Y}$ denotes the training data with image $x_i\in\mathscr{X},\forall i$ and its class label $y_i\in\mathscr{Y}$, $\omega\in\Omega$ denotes the parameters for the network defined by function $\phi:\mathscr{X}\times\Omega\to\mathscr{Y}$, $\theta\in\Theta$ denotes the templates in the feasible space Θ that ω should match with, $\ell:\mathscr{Y}\times\mathscr{Y}\to\mathbb{R}$ denotes the loss function, $\mathscr{R}_1$ denotes the weight decay with ℓ_2 norm, $\mathscr{R}_2:\Omega\times\Theta\to\mathbb{R}$ denotes the regularizer that measures the difference between ω and θ, and $\lambda_1,\lambda_2\geq 0$ are predefined constants. Different from conventional CNNs, here we propose learning not only the network weights ω but also the matching templates θ within the feasible space Θ that encodes certain constraints on the templates such as symmetry. In the sequel we will explain how to effectively design a scale-invariant network ϕ, and how to efficiently construct a rotation-invariant regularizer $\mathscr{R}_2$.

6.4.1 Network Architecture

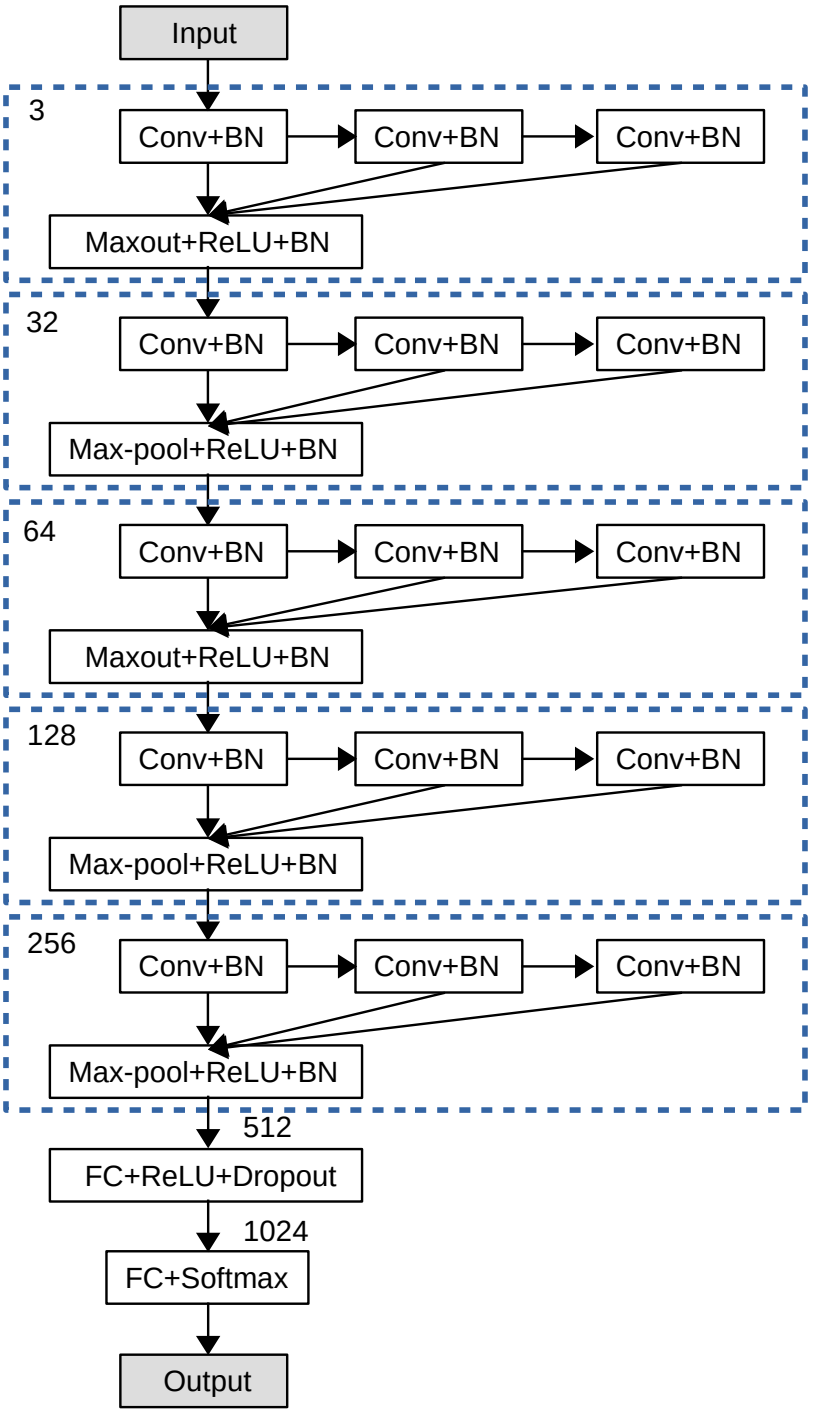


Figure 6.2: Illustration of the network we use in our experiments for learning affine-invariant features. Each dashed block is a multi-scale maxout block accounting for scale invariance, and the numbers here denote the default dimensions of inputs for the corresponding blocks and layers.

We illustrate our network in Fig. 6.2, where all the operations are basic and widely used in CNNs such as batch normalization (BN) (57), and "+" denotes one operation followed by the other. Due to the small image sizes (*e.g.* 32×32 pixels) in our experiments, we conduct downsampling for three times only using max-pooling. In each block the first convolutional layer is responsible for mapping the inputs into a higher dimensional space, *e.g.* $3 \rightarrow 32$, and the other two convolutional layers learn the (linear) transformation in the same space, *e.g.* $32 \rightarrow 32$. For grayscale images, the input dimension is changed from 3 to 1.

Different from existing networks such as GoogLeNet and TI-Pooling, we propose extracting features within different scales using a sequence of convolutional operations. Considering the trade-off between computational efficiency and accuracy, we only exploit three scales, *i.e.* $3 \times 3, 5 \times 5, 7 \times 7$, using fixed filter size of 3×3 in each convolutional layer, and use maxout to select a scale with the maximum response. This scale is taken as the best one to fit for the object. In fact we use two and three 3×3 convolutions to approximate the responses with filter sizes of 5×5 and 7×7, respectively, for efficient computation. With the increase of the network depth, information within larger scales (*i.e.* receptive field) can be extracted as well.

We also find that the network depth is more important than the network width *w.r.t.* the accuracy. It has been demonstrated in Wide Residual Networks (WRN) (162) that wider networks can improve the performance. In contrast to the parallel mechanism in WRN, in each block we apply convolutions sequentially. Note that the proposed mechanism can be integrated with other networks as well.

6.4.2 Training with Rotation-Invariant Regularizer

6.4.2.1 General Formulation

As illustrated in Fig. 6.1(b), in order to enforce the filters to satisfy certain spatial properties such as rotation invariance, the templates here need to be constructed in certain way to encode such properties. Therefore, we propose the following general formulation for rotation-invariant

(a) (b)

Figure 6.3: Examples of weight patterns, defined by hash function h, that can be used to approximate circular patterns for rotation invariance. In each subfigure the same color denotes the same weight.

regularizers:

$$\begin{aligned} &\mathscr{R}_2(\omega,\theta) \\ &= \mathbb{E}_{k\sim\mathscr{K}}\left[\sum_{m=-p_k}^{p_k}\sum_{n=-q_k}^{q_k} d\Big(\omega_k(m,n),\theta_k(h(m,n))\Big)\right], \end{aligned} \tag{6.3}$$

where $k \in \mathscr{K}$ denotes the index of a 2D spatial filter, $\mathbb{E}_{k\sim\mathscr{K}}$ denotes the expectation over all 2D spatial filters, (m,n) denotes the 2D-index of a weight in the k-th filter with size (M_k,N_k), $p_k = \left\lceil \frac{M_k}{2} \right\rceil, q_k = \left\lceil \frac{N_k}{2} \right\rceil$, $\lceil\cdot\rceil$ denotes the ceiling function, $d : \mathbb{R}\times\mathbb{R} \to \mathbb{R}$ denotes a distance function, $h : \mathbb{R}\times\mathbb{R} \to \mathbb{R}$ denotes a hash function that determines the weight pattern in the templates for matching, and correspondingly $\theta : \mathbb{R} \to \mathbb{R}$ is a learnable function.

Choices of Distance Function d. In general we do not have any explicit requirement on d. For instance, it can be ℓ_1-norm, ℓ_2-norm, or group sparsity norm such as $\ell_{2,1}$-norm. Moreover, this distance measure can be conducted in not only Euclidean but also non-Euclidean spaces such as manifold regularization (5), which will be appreciated in geometric deep learning (11).

Choices of Hash Function h. For rotation invariance, ideally it should be a circular pattern defined by $h(m,n) = (m^2+n^2)^{\frac{1}{2}}$ in a continuous space. Due to the discretization of images, however, it hardly forms circles in filters without interpolation which will significantly increase the computational complexity in convolution. Instead, we propose learning some simpler patterns that can be used to approximate circles. For instance, we illustrate two exemplar patterns for filters with

size 3×3 in Fig. 6.3, where the patterns in (a) and (b) are defined by $h(m,n) = \left\lfloor (m^2+n^2)^{\frac{1}{2}} \right\rfloor$ and $h(m,n) = \left\lceil (m^2+n^2)^{\frac{1}{2}} \right\rceil$, respectively, and $\lfloor \cdot \rfloor$ is the floor function. Other hash functions may be also applicable here, but finding the best one is outside the scope of this paper.

6.4.2.2 An Empirical Showcase

In this section we will show a specific regularizer that we use in our experiments later. For the simplicity and efficiency, we decide to employ the least square loss for d and the pattern in Fig. 6.3(a) for h without fine-tuning the accuracy on the data sets.

Specifically we define our empirical rotation-invariant regularizer as follows:

$$\begin{aligned} &\mathscr{R}_2(\omega, \theta) \\ &= \mathbb{E}_{k \sim \mathscr{K}} \left[\sum_{m,n \neq 0} \left(\omega_k(m,n) - \frac{\sum_{m',n' \neq 0} \omega_k(m',n')}{p_k q_k - 1} \right)^2 \right], \end{aligned} \tag{6.4}$$

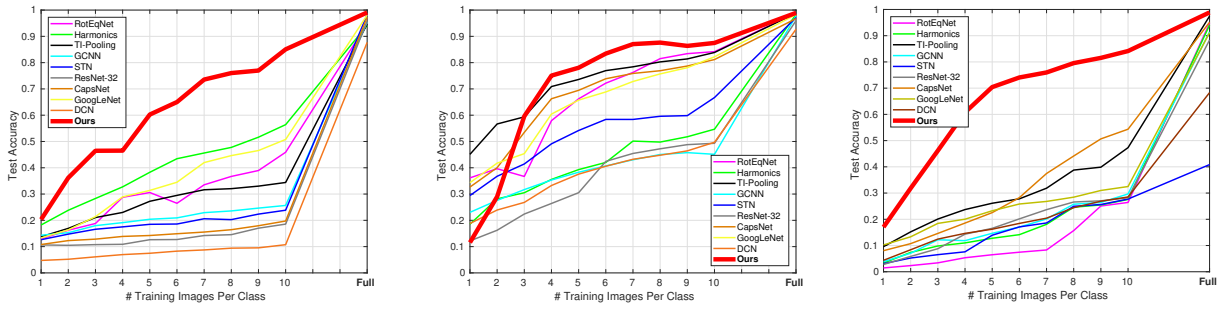
where $\theta_k(h(m,n)) = \frac{\sum_{m',n' \neq 0} \omega_k(m',n')}{p_k q_k - 1}$ is a scalar.

Similar to the center loss in (142), here we aims to reduce the variance among the weights in each 2D spatial filter with 3×3 pixels, on average. Meanwhile, the patterns in the templates are updated automatically with the mean of the weights. In this way we can learn filters that can better approximate 2D spatial circular patterns for rotation invariance. In backpropagation, since $\mathscr{R}_2(\omega, \theta)$ in Eq. 6.4 is always differentiable *w.r.t.* $\omega_k, \forall k$, any deep learning solver such as stochastic gradient descent (SGD) can be used to train the network with our rotation-invariant regularizer.

Discussion. Recall that Fig. 6.3 essentially encodes the structural patterns that we expect for learned filters to handle rotation. One may argue that we can enforce such structures into learning strictly by converting the regularizer $\mathscr{R}_2$ in Eq. 6.2 into constraints and solving a constrained nonconvex optimization problem. We decide not to do so because potentially the new problem will be much harder to be solved than the one in Eq. 6.2. Besides since the structures in Fig. 6.3 are already the approximation of the circular structure, we do not necessarily guarantee that all

	Ours	RotEqNet	Harmonics	TI-Pooling	GCNN	STN	ResNet-32	CapsNet	GoogLeNet	DCN
aff. (F)	**99.08**	94.81	94.20	94.72	95.43	98.24	95.76	97.30	98.12	87.70
rot (F)	**98.92**	98.91	98.31	98.80	97.72	97.12	95.96	96.73	98.29	92.69
T. S. (F)	**98.87**	94.79	94.02	97.47	91.47	40.87	88.35	95.15	91.16	68.29
Ave. (F)	**98.95**	96.17	95.51	96.99	94.87	78.74	93.36	96.39	95.85	82.75
aff. (10)	**85.06**	45.91	56.41	34.40	25.67	23.85	18.56	19.74	50.77	10.74
rot (10)	**87.49**	84.18	54.67	83.86	45.12	66.72	49.31	81.17	82.20	49.69
T. S. (10)	**84.15**	26.43	27.57	47.31	29.66	27.72	28.20	54.35	32.49	28.52
Ave. (10)	**85.56**	52.17	46.21	55.19	33.48	39.43	32.02	51.75	55.15	29.65

Table 6.1: Test accuracy (%) comparison on different datasets under two training settings: **(F)** with all the images, and **(10)** with 10 random images per class.



(a) affNIST (b) MNIST-rot (c) Traffic Sign

Figure 6.4: Test accuracy comparison of different networks on the three data sets. "Full" here indicates that we use all the training images. Our approach significantly outperforms the state-of-the-art, especially with small numbers of training images.

the weights with the same color are identical. More freedom as in regularization may lead to a compensation for the loss of the structural approximation in terms of accuracy.

6.5 Experiments

6.5.1 Benchmark Data with Affine Transformations

6.5.1.1 Experimental Setup

Data Sets. We test our approach on three benchmark data sets, affNIST (107), MNIST-rot (74), and Traffic Sign (124).

affNIST is created by applying random small affine transformations to each 28×28 grayscale image in MNIST (LECUN) (10 classes). It is designed for testing the tolerance of an algorithm

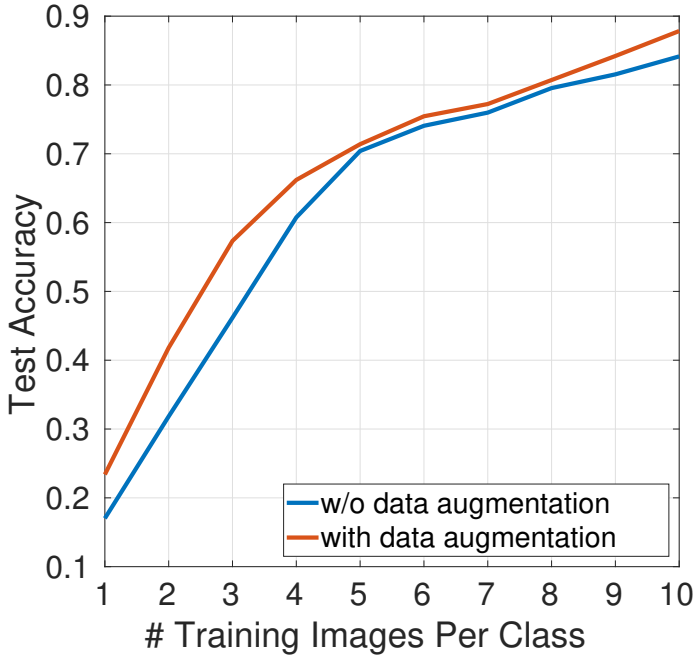


Figure 6.5: Data augmentation comparison on Traffic Sign.

to such transformations. There are 60K training and validation samples and 10K test samples in affNIST with size 40×40 pixels. To facilitate the data processing in training, we resize all the images to 32×32 pixels.

MNIST-rot (74) is another variant of MNIST, where a random rotation between 0° and 360° is applied to each image. It has 10K/2K/50K training/validation/test samples. To facilitate the data processing in training, we again resize all the grayscale images to 32×32 pixels.

Traffic Sign contains 43 classes with unbalanced class frequencies, 34799 training RGB images, and 12630 testing RGB images with size of 32×32 pixels. It reflects the strong variations in visual appearance of signs due to distance, illumination, weather conditions, partial occlusions, and rotations, leading to a very challenging recognition problem.

Networks. We compare our approach with some state-of-the-art networks with similar model complexity to ours, *i.e.* RotEqNet (96)[1], Harmonics (144)[2], TI-Pooling (73)[3], GCNN (20)[4], STN (59)[5],

[1] https://github.com/COGMAR/RotEqNet
[2] https://github.com/deworrall92/harmonicConvolutions
[3] https://github.com/dlaptev/TI-pooling
[4] https://github.com/tscohen/gconv_experiments
[5] https://github.com/kevinzakka/spatial-transformer-network

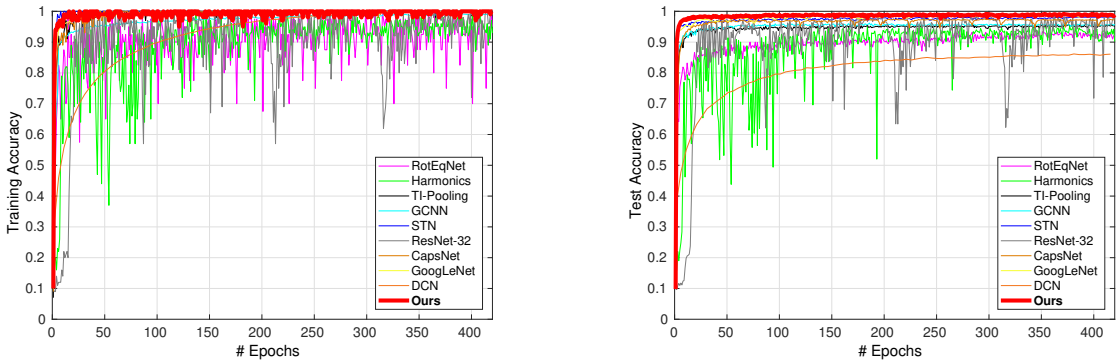


Figure 6.6: Illustration of training/testing behavior of different networks on affNIST.

ResNet-32 (45)[6], CapsNet (107)[7], GoogLeNet (125)[8], and DCN (23)[9]. Specifically TI-Pooling is designed for scale invariance, RotEqNet, Harmonics, and GCNN are designed for rotation invariance. We use the public code for our comparison.

We implement our default network using Tensorflow and following the architecture in Fig. 6.2 with the default numbers of channels. Note that the implementation of the networks in our comparison may be different, (*i.e.* GCNN→Chainer; GoogLeNet, DCN→Keras; CapsNet, TI-Pooling, Harmonics, STN, ResNet-32→Tensorflow; RotEqNet→Pytorch) which may lead to various computational efficiency.

Training Protocols. We tune each network to report its best performance on the data sets. By default we train the networks for 42000 iterations with mini-batch size 100, weight decay $\lambda_1 = 0.0005$, and momentum 0.9. The global learning rate is set to 0.01 or 0.0001 when trained using all or a few training images per class, respectively, and it is reduced by 0.1 twice at the 20000 iteration and the 30000 iteration as well. For each network the hyper-parameter tuning starts with the default setting, and the best setting may be slightly different from the default. We follow this default setting in all the experiments and set $\lambda_2 = 150$. The numbers reported here are the average over three trials.

To do fair comparison, we follow the settings for data augmentation in the publications of most

[6] https://github.com/tensorflow/models/tree/master/research/resnet
[7] https://github.com/naturomics/CapsNet-Tensorflow
[8] https://github.com/flyyufelix/cnn_finetune/blob/master/googlenet
[9] https://github.com/felixlaumon/deform-conv

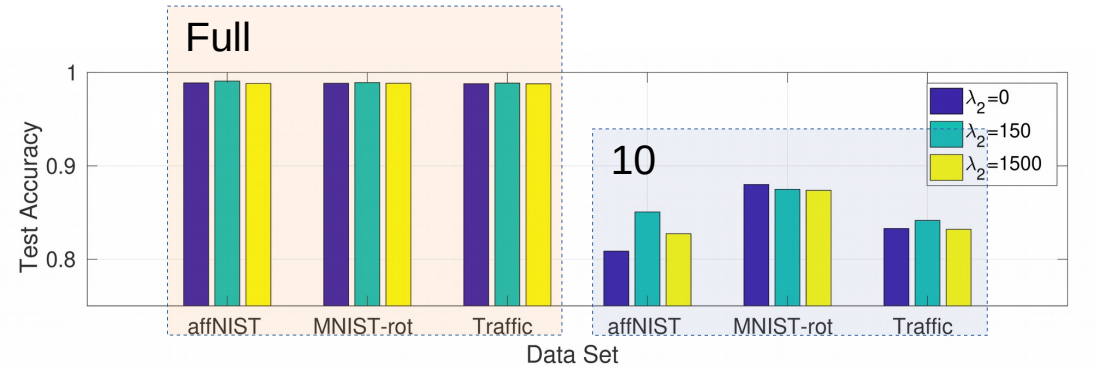
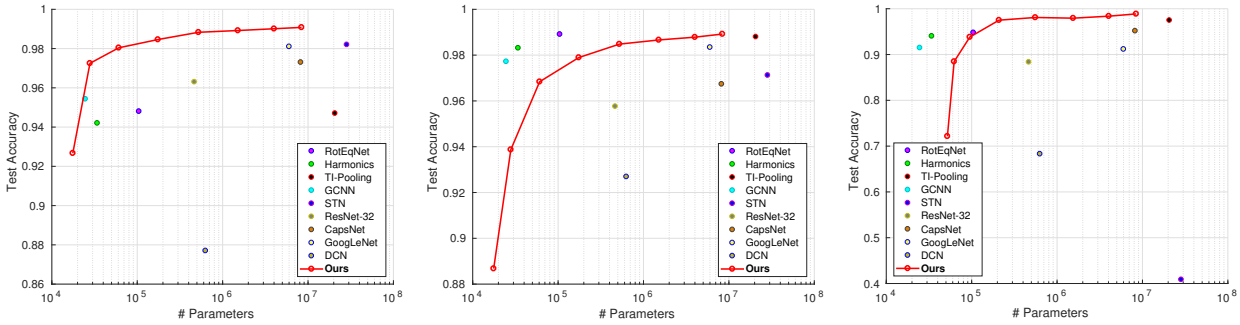


Figure 6.7: Illustration of the effect of λ_2 in Eq. 6.2 on test accuracy.



(a) affNIST (b) MNIST-rot (c) Traffic Sign

Figure 6.8: Test accuracy comparison with the others using different numbers of parameters. Best viewed in color.

of the competitors. Specifically, by default on affNIST and Traffic Sign we do not employ data augmentation, but on MNIST-rot we do.

6.5.1.2 Results

Better Generalization, Data-Efficiency, & Robustness. We summarize the test accuracy comparison in Table 6.1. As we see, using either all or 10 random training/validation images per class, our method consistently outperforms the competitors on the three data sets with a margin of **1.96%** or **30.37%**. Using the full set the stds of all the methods are small and similar, and thus we do not show the numbers.

To better demonstrate the data-efficiency, we illustrate test accuracy comparison using few random training/validation images per class in Fig. 6.4. Overall, our method works significantly better than the competitors with large margins. Note that on MNIST-rot our performance is worse

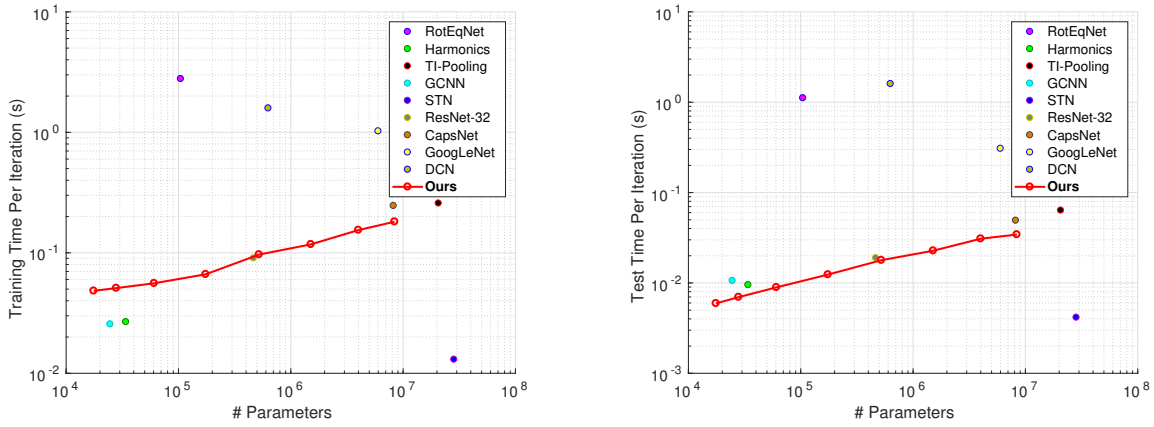


Figure 6.9: Training/Test time comparison with the others using different numbers of parameters. Best viewed in color.

than some of the competitors when using 1 or 2 images per class for training. A possible reason may come from data augmentation. Another reason is that some of the networks are designed specifically for rotation invariance and this data set just fits for this purpose. With the increase of the numbers of training samples, however, our method again beats all the competitors. It is worth mentioning that in Harmonics Networks (144), similar experiments on MNIST-rot were conducted to show data-efficiency and robustness of the approach. Using $\frac{1}{6}$ of the full training/validation data Harmonics lost about 3%. Here we compare different networks using less than $\mathbf{\frac{1}{120}}$ to show the superiority of our method over the others. Empirically we observe that our method can work very robustly with standard deviation of less than 1%, in general.

In addition, we can further improve our performance using data augmentation. In Fig. 6.5 we illustrate the performance comparison on Traffic Sign with or without data augmentation. As we see, using 10 random training images per class we can achieve 87.84% with 3.69% improvement.

Training & Testing Behavior. We illustrate the training and test accuracy behavior of each network on affNIST with the full training set in Fig. 6.6. As we see all the networks are well trained with convergence. In the testing stage our network converge faster than most of the competitors with better accuracy. Similar observations can be made in training as well. We make similar observations on the other two data sets. From this perspective, we can also demonstrate that our method has better generalization.

	2×[Conv+BN]	3×[Conv+BN]	4×[Conv+BN]
affNIST (F)	99.04	99.08	98.69
MNIST-rot (F)	98.72	98.92	98.97
Traffic Sign (F)	98.42	98.87	98.42
Average	98.73	98.95	98.69

Table 6.2: Effect on test accuracy (%) of different multi-scale settings, where our default setting is 3×[Conv+BN].

Effect of Multi-Scale Maxout. In Table 6.2 we list the test accuracy using different multi-scale settings, while fixing the parameter $\lambda_2 = 150$. As we see the changes between different settings are really marginal, which again demonstrates the good generalization and robustness of our method. Considering the trade-off between accuracy and computational efficiency, we choose 3×[Conv+BN] as our default setting used in Fig. 6.2.

Effect of Rotation-Invariant Regularization. We illustrate such effect in Fig. 6.7 while using the default multi-scale maxout setting. With different values where $\lambda_2 = 0$ means no our regularizer, we can see that using the full set for training our performances are almost identical. This is probably because the number of training images is sufficiently large to capture the scaling and rotation information already. Using a few training images, *e.g.* 10 per class, the benefit of using our rotation-invariant regularizer becomes much clearer, especially on affNIST. Using $\lambda_2 = 150$ as default, there is 1.52%, on average, improvement over that without our regularizer.

We also observe that our rotation-invariant regularizer can achieve very small numbers empirically. For instance, on affNIST the value is 2.94×10^{-7}, indicating that our learned filters are very close to the spatial circular patterns.

Behavior with Different Numbers of Parameters. We reduce the number of parameters in our network by channel-wise shrinking. Specifically in ascending order of number of parameters, the corresponding network channels are set as follows: [4,4,4,4,4], [16,16,16,16,16], [32,32,32,32,32], [32,64,64,64,64], [32,64,128,128,128], [32,64,128,256,256], [32,64,128,256,512], followed by an FC of 1024 nodes and another FC for classification.

We first compare our performance using different numbers with the competitors in Fig. 6.8. We can see that after about 200K parameters the improvement of our approach becomes slow, while

before 200K our performance drops significantly with the decrease of numbers of parameters. In the figure 200K corresponds to the setting [32,64,64,64,64], whose performance is, or on par with, the best already.

We then compare the running time per iteration in both training and testing stages in Fig. 6.9. We run all the code on the same machine with a Titan XP GPU. In training the running time includes the feedforward calculation and backpropagation inference (dominating training time), while in testing the running time only includes the feedforward calculation. As we see, in both training and testing our computational complexity grows exponentially, in general, with the number of parameters (note that the y-axis is in log-scale). Although some codes are written in different deep learning environments, we can still do a fair comparison with Harmonics and STN. Harmonics has fewer parameters, leading to faster backpropagation and thus shorter training time. The operations in Harmonic, however, is more complex than ours, and thus with a similar number of parameters our method is faster in testing. The operations in STN are much simpler than both Harmonics and ours, leading to faster running speed in both training and testing. Note that in order to further improve our computational efficiency, we can simply remove one Conv+BN in the multi-scale maxout block that can achieve similar accuracy (see Table 6.2).

6.5.2 Comparison on CIFAR-100 (Krizhevsky et al.)

Beyond the benchmark data sets with affine transformations, we also test our method on "natural" images. For instance, we illustrate our comparison results on CIFAR-100 in Fig. 6.10. CIFAR-100 contains 60,000 32×32 color images in 100 different classes, 500/100 training/testing images per class. Following the same training protocol, we randomly sample a few images per class to further demonstrate our superiority, especially on data-efficiency.

As we see in Fig. 6.10, our method significantly and consistently outperforms the competitors with a few training samples. For instance, using 100 samples per class ours achieves 52.67% test accuracy with the improvement of almost 10% over ResNet-32 (the second best). Using the full training set, ours achieves 78.33% that is slightly lower than WRN-28-10 (80.75%), but higher than

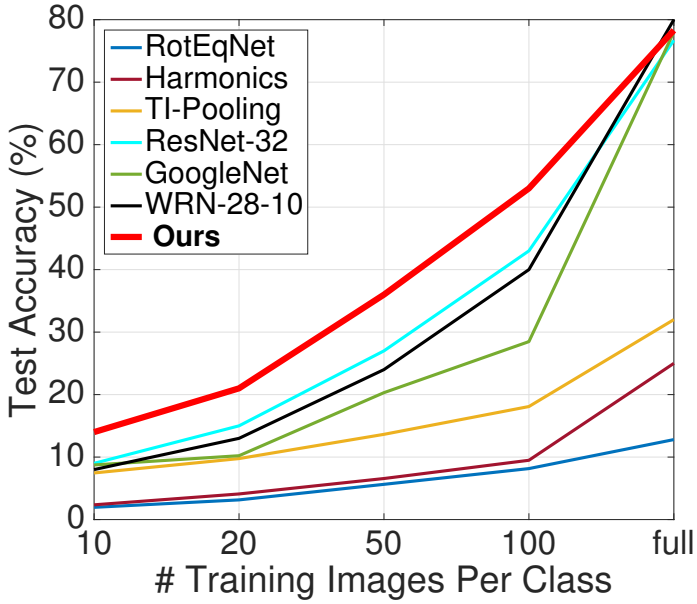


Figure 6.10: Test accuracy comparison of different networks on CIFAR-100. "Full" here indicates that we use all the training images. Again our approach significantly outperforms the state-of-the-art, especially with small numbers of training images.

ResNet-32 (76.7%) and GoogleNet (78.03%), and dramatically higher than the other networks that learn the scale or rotation invariant representations such as TI-Pooling (31.77%).

6.6 Conclusion

In this paper we propose a novel multi-scale maxout deep CNN and a novel rotation-invariant regularizer to learn affine-invariant representations for object recognition in images. Multi-scale convolution with maxout can handle translation and scale, and enforcing 2D filters to approximate circular patterns by our regularization can manage to induce invariance to rotation. By taking these as a priori knowledge, we can easily interpret our network architecture as well as its training procedure. We test our method on three benchmark data sets as well as CIFAR-100 to demonstrate its superiority over the state-of-the-art in terms of generalization, data-efficiency, and robustness. Especially, with a few training samples our method can work significantly better, leading to the hypothesis that the introduction of a priori knowledge into deep learning can effectively reduce the amount of data to accomplish the tasks.

Chapter 7

Conclusion and Future Work

Efficiently and effectively Extracting meaningful information from data is crucial in nowadays' research. Inspired by how animals behavior, we study different data processing methods to assistant autonomous systems for a better understanding of the environment. Furthermore, we prefer deep neural network to learn the knowledge from a significant amount of data. In summary, we explore a optimization based sensor fusion method, which combine the IMU data and image data for pose estimation and 3D reconstruction; For the following work, we will focus on design a deep neural network model working as a pilot. The goal of this model is to learn a mapping between the input sensor data and the control command. To this end, we will first collect a large amount of flight data recording the state of the UAV and the corresponding control command. Then we train a well designed neural network model on this dataset so that this model learns to make prediction of the needed control command. Validation and Evaluation of the effectiveness will be done on the trained model. Finally, we hope to achieve a deep neural network based pilot that generate control command to control a UAV.

References

[1] Al-Sharman, M. K., Zweiri, Y., Jaradat, M. A. K., Al-Husari, R., Gan, D., & Seneviratne, L. D. (2019). Deep-learning-based neural network training for state estimation enhancement: application to attitude estimation. *IEEE Transactions on Instrumentation and Measurement*.

[2] Amos, B., Jimenez, I., Sacks, J., Boots, B., & Kolter, J. Z. (2018). Differentiable mpc for end-to-end planning and control. In *NIPS* (pp. 8299–8310).

[3] Andreas, J., Rohrbach, M., Darrell, T., & Klein, D. (2016). Neural module networks. In *CVPR* (pp. 39–48).

[4] Audebert, N., Le Saux, B., & Lefèvre, S. (2016). Semantic segmentation of earth observation data using multimodal and multi-scale deep networks. In *ACCV* (pp. 180–196).

[5] Belkin, M., Niyogi, P., & Sindhwani, V. (2006). Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *JMLR*, 7(Nov), 2399–2434.

[6] Bergamo, A., Bazzani, L., Anguelov, D., & Torresani, L. (2014). Self-taught object localization with deep networks. *arXiv*.

[7] Bilen, H., Pedersoli, M., & Tuytelaars, T. (2015). Weakly supervised object detection with convex clustering. In *CVPR*.

[8] Bilen, H. & Vedaldi, A. (2015). Weakly supervised deep detection networks. *arXiv preprint arXiv:1511.02853*.

[9] Bodla, N., Singh, B., Chellappa, R., & Davis, L. S. (2017). Improving object detection with one line of code. *arXiv preprint arXiv:1704.04503*.

[10] Brahmbhatt, S., Gu, J., Kim, K., Hays, J., & Kautz, J. (2018). Geometry-aware learning of maps for camera localization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[11] Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., & Vandergheynst, P. (2017). Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4), 18–42.

[12] Burri, M., Nikolic, J., Gohl, P., Schneider, T., Rehder, J., Omari, S., Achtelik, M. W., & Siegwart, R. (2016). The EuRoC micro aerial vehicle datasets. *The International Journal of Robotics Research*, 35(10), 1157–1163.

[13] Carrio, A., Bavle, H., & Campoy, P. (2018). Attitude estimation using horizon detection in thermal images. *International Journal of Micro Air Vehicles*, 10(4), 352–361.

[14] Cen, F. & Wang, G. (2019). Dictionary representation of deep features for occlusion-robust face recognition. *IEEE Access*.

[15] Chen, S. (2011). Kalman filter for robot vision: a survey. *IEEE Transactions on industrial electronics*, 59(11), 4409–4420.

[16] Chu, X., Yang, W., Ouyang, W., Ma, C., Yuille, A. L., & Wang, X. (2017). Multi-context attention for human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1831–1840).

[17] Cinbis, R. G., Verbeek, J., & Schmid, C. (2014). Multi-fold mil training for weakly supervised object localization. In *CVPR*.

[18] Cinbis, R. G., Verbeek, J., & Schmid, C. (2017). Weakly supervised object localization with multi-fold multiple instance learning. *IEEE transactions on pattern analysis and machine intelligence*, 39(1).

[19] Clark, R., Wang, S., Markham, A., Trigoni, N., & Wen, H. (2017). Vidloc: A deep spatio-temporal model for 6-dof video-clip relocalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 6856–6864).

[20] Cohen, T. & Welling, M. (2016). Group equivariant convolutional networks. In *ICML* (pp. 2990–2999).

[21] Concha, A., Loianno, G., Kumar, V., & Civera, J. (2016). Visual-inertial direct SLAM. In *Proceedings of IEEE International Conference on Robotics and Automation* (pp. 1331–1338).

[22] Crandall, D., Felzenszwalb, P., & Huttenlocher, D. (2005). Spatial priors for part-based recognition using statistical models. In *CVPR*, volume 1 (pp. 10–17).

[23] Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., & Wei, Y. (2017). Deformable convolutional networks. In *CVPR* (pp. 764–773).

[24] de Avila Belbute-Peres, F., Smith, K., Allen, K., Tenenbaum, J., & Kolter, J. Z. (2018). End-to-end differentiable physics for learning and control. In *NIPS* (pp. 7178–7189).

[25] Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *CVPR*: IEEE.

[26] Deselaers, T., Alexe, B., & Ferrari, V. (2012). Weakly supervised localization and learning with generic knowledge. *IJCV*, 100(3).

[27] Ellingson, G., Wingate, D., & McLain, T. (2017). Deep visual gravity vector detection for unmanned aircraft attitude estimation. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 5557–5563).: IEEE.

[28] Engel, J., Koltun, V., & Cremers, D. (2016). Direct sparse odometry. *arXiv preprint arXiv:1607.02565*.

[29] Engel, J., Schöps, T., & Cremers, D. (2014). Lsd-slam: Large-scale direct monocular slam. In *European Conference on Computer Vision* (pp. 834–849).: Springer.

[30] Engel, J., Schöps, T., & Cremers, D. (2015a). LSD-SLAM: Large-scale direct monocular SLAM. In *Proceedings of European Conference on Computer Vision* (pp. 834–849).: Springer.

[31] Engel, J., Stückler, J., & Cremers, D. (2015b). Large-scale direct SLAM with stereo cameras. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robot Systems*.

[32] Engel, J., Sturm, J., & Cremers, D. (2013a). Semi-dense visual odometry for a monocular camera. In *Proceedings of IEEE International Conference on Computer Vision*.

[33] Engel, J., Sturm, J., & Cremers, D. (2013b). Semi-dense visual odometry for a monocular camera. In *Proceedings of the IEEE international conference on computer vision* (pp. 1449–1456).

[34] Everingham, M., Eslami, S. A., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2015). The pascal visual object classes challenge: A retrospective. *IJCV*, 111(1).

[35] Felzenszwalb, P. F. & Huttenlocher, D. P. (2012). Distance transforms of sampled functions. *Theory Of Computing*, 8, 415–428.

[36] Fischler, M. A. & Elschlager, R. A. (1973). The representation and matching of pictorial structures. *IEEE Transactions on computers*, 100(1), 67–92.

[37] Forster, C., Carlone, L., Dellaert, F., & Scaramuzza, D. (2015). IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation. In *Robotics: Science and Systems*: Georgia Institute of Technology.

[38] Forster, C., Carlone, L., Dellaert, F., & Scaramuzza, D. (2016). On-manifold preintegration for real-time visual–inertial odometry. *IEEE Transactions on Robotics*, PP(99), 1–21.

[39] Forster, C., Pizzoli, M., & Scaramuzza, D. (2014). Svo: Fast semi-direct monocular visual odometry. In *2014 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 15–22).: IEEE.

[40] Girshick, R. (2015). Fast r-cnn. In *ICCV*.

[41] Girshick, R., Iandola, F., Darrell, T., & Malik, J. (2015). Deformable part models are convolutional neural networks. In *CVPR* (pp. 437–446).

[42] Gokberk Cinbis, R., Verbeek, J., & Schmid, C. (2014). Multi-fold mil training for weakly supervised object localization. In *CVPR*.

[43] Goodfellow, I. J., Warde-Farley, D., Mirza, M., Courville, A., & Bengio, Y. (2013). Maxout networks. In *ICML* (pp. III–1319).

[44] Guillaumin, M. & Ferrari, V. (2012). Large-scale knowledge transfer for object localization in imagenet. In *CVPR*.

[45] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778).

[46] He, L., Wang, G., & Hu, Z. (2018a). Learning depth from single images with deep neural network embedding focal length. *IEEE Transactions on Image Processing*, 27(9), 4676–4689.

[47] He, L., Yu, M., & Wang, G. (2018b). Spindle-net: Cnns for monocular depth inference with dilation kernel method. In *2018 24th International Conference on Pattern Recognition (ICPR)* (pp. 2504–2509).: IEEE.

[Hinton] Hinton, G. Taking inverse graphics seriously. https://www.cs.toronto.edu/ hinton/csc2535/notes/lec6b.pdf.

[49] Hinton, G., Frosst, N., & Sabour, S. (2018). Matrix capsules with em routing. In *ICLR*.

[50] Hinton, G. E., Krizhevsky, A., & Wang, S. D. (2011). Transforming auto-encoders. In *International Conference on Artificial Neural Networks* (pp. 44–51).: Springer.

[51] Hoffman, J., Guadarrama, S., Tzeng, E. S., Hu, R., Donahue, J., Girshick, R., Darrell, T., & Saenko, K. (2014). Lsda: Large scale detection through adaptation. In *NIPS*.

[52] Hong, C., Chen, X., Wang, X., & Tang, C. (2016). Hypergraph regularized autoencoder for image-based 3d human pose recovery. *Signal Processing*, 124, 132–140.

[53] Hong, C., Yu, J., Wan, J., Tao, D., & Wang, M. (2015). Multimodal deep autoencoder for human pose recovery. *IEEE Transactions on Image Processing*, 24(12), 5659–5670.

[54] Hoogeboom, E., Peters, J. W., Cohen, T. S., & Welling, M. (2018). Hexaconv. *arXiv preprint arXiv:1803.02108*.

[55] Huang, G., Chen, D., Li, T., Wu, F., van der Maaten, L., & Weinberger, K. Q. (2017). Multi-scale dense networks for resource efficient image classification. *arXiv preprint arXiv:1703.09844*.

[56] Huang, S. & Dissanayake, G. (2007). Convergence and consistency analysis for extended kalman filter based slam. *IEEE Transactions on robotics*, 23(5), 1036–1049.

[57] Ioffe, S. & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML* (pp. 448–456).

[58] Isik, L., Meyers, E. M., Leibo, J. Z., & Poggio, T. (2013). The dynamics of invariant object recognition in the human visual system. *Journal of neurophysiology*, 111(1), 91–102.

[59] Jaderberg, M., Simonyan, K., Zisserman, A., et al. (2015). Spatial transformer networks. In *NIPS* (pp. 2017–2025).

[60] Jia, X., De Brabandere, B., Tuytelaars, T., & Gool, L. V. (2016). Dynamic filter networks. In *NIPS* (pp. 667–675).

[61] Jie, Z., Wei, Y., Jin, X., Feng, J., & Liu, W. (2017). Deep self-taught learning for weakly supervised object localization. In *CVPR*.

[62] Kanazawa, A., Sharma, A., & Jacobs, D. (2014). Locally scale-invariant convolutional neural networks. *arXiv preprint arXiv:1412.5104*.

[63] Kendall, A. & Cipolla, R. (2017). Geometric loss functions for camera pose regression with deep learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 5974–5983).

[64] Kendall, A., Grimes, M., & Cipolla, R. (2015). Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE international conference on computer vision* (pp. 2938–2946).

[65] Kerl, C., Sturm, J., & Cremers, D. (2013). Dense visual SLAM for RGB-D cameras. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robot Systems*.

[66] Kingma, D. P. & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

[Krizhevsky et al.] Krizhevsky, A., Nair, V., & Hinton, G. Cifar-100 (canadian institute for advanced research).

[68] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *NIPS*.

[69] Kuen, J., Wang, Z., & Wang, G. (2016). Recurrent attentional networks for saliency detection. In *Proceedings of the IEEE Conference on computer Vision and Pattern Recognition* (pp. 3668–3677).

[70] Kuse, M. P. & Shen, S. (2016). Robust camera motion estimation using direct edge alignment and sub-gradient method. In *IEEE International Conference on Robotics and Automation (ICRA-2016), Stockholm, Sweden*.

[71] Lai, B. & Gong, X. (2017). Saliency guided end-to-end learning for weakly supervised object detection. *arXiv preprint arXiv:1706.06768*.

[72] Lapin, M., Hein, M., & Schiele, B. (2014). Learning using privileged information: Svm+ and weighted svm. *Neural Networks*, 53.

[73] Laptev, D., Savinov, N., Buhmann, J. M., & Pollefeys, M. (2016). Ti-pooling: transformation-invariant pooling for feature learning in convolutional neural networks. In *CVPR* (pp. 289–297).

[74] Larochelle, H., Erhan, D., Courville, A., Bergstra, J., & Bengio, Y. (2007). An empirical evaluation of deep architectures on problems with many factors of variation. In *ICML* (pp. 473–480).

[75] Larsson, G., Maire, M., & Shakhnarovich, G. (2016). Fractalnet: Ultra-deep neural networks without residuals. *arXiv preprint arXiv:1605.07648*.

[76] Laskar, Z., Melekhov, I., Kalia, S., & Kannala, J. (2017). Camera relocalization by computing pairwise relative poses using convolutional neural network. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 929–938).

[LECUN] LECUN, Y. The mnist database of handwritten digits. *http://yann.lecun.com/exdb/mnist/*.

[78] Leutenegger, S., Lynen, S., Bosse, M., Siegwart, R., & Furgale, P. (2015). Keyframe-based visual–inertial odometry using nonlinear optimization. volume 34 (pp. 314–334).: Sage.

[79] Li, D., Huang, J.-B., Li, Y., Wang, S., & Yang, M.-H. (2016). Weakly supervised object localization with progressive domain adaptation. In *CVPR*.

[80] Li, M. & Mourikis, A. I. (2012). Improving the accuracy of ekf-based visual-inertial odometry. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on* (pp. 828–835).: IEEE.

[81] Li, M. & Mourikis, A. I. (2013). High-precision, consistent EKF-based visual–inertial odometry. *International Journal of Robotics Research*, 32(6), 690–711.

[82] Li, W., Zhu, X., & Gong, S. (2018). Harmonious attention network for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2285–2294).

[83] Liang, H.-J., Sanket, N. J., Fermüller, C., & Aloimonos, Y. (2019). Salientdso: Bringing attention to direct sparse odometry. *IEEE Transactions on Automation Science and Engineering*.

[84] Liao, Z. & Carneiro, G. (2015). Competitive multi-scale convolution. *arXiv preprint arXiv:1511.05635*.

[85] Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. In *CVPR* (pp. 2117–2125).

[86] Ling, Y., Liu, T., & Shen, S. (2016). Aggressive quadrotor flight using dense visual-inertial fusion. In *Proceedings of IEEE International Conference on Robotics and Automation*.

[87] Ling, Y. & Shen, S. (2015). Dense visual-inertial odometry for tracking of aggressive motions. In *Proceedings of IEEE International Conference on Robotics and Biomimetics*.

[88] Liu, X., Song, M., Zhang, L., Tao, D., Bu, J., & Chen, C. (2012). Pedestrian detection using a mixture mask model. In *Proceedings of 2012 9th IEEE International Conference on Networking, Sensing and Control* (pp. 271–276).: IEEE.

[89] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2), 91–110.

[90] Lu, Y. & Song, D. (2015). Robust rgb-d odometry using point and line features. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 3934–3942).

[91] Luan, S., Chen, C., Zhang, B., Han, J., & Liu, J. (2018). Gabor convolutional networks. *TIP*.

[92] Lupton, T. & Sukkarieh, S. (2012). Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions. *IEEE Transactions on Robotics*, 28(1), 61–76.

[93] Lynen, S., Achtelik, M. W., Weiss, S., Chli, M., & Siegwart, R. (2013). A robust and modular multi-sensor fusion approach applied to mav navigation. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robot Systems*.

[94] Ma, W., Wu, Y., Wang, Z., & Wang, G. (2018). Mdcn: Multi-scale, deep inception convolutional neural networks for efficient object detection. In *2018 24th International Conference on Pattern Recognition (ICPR)* (pp. 2510–2515).: IEEE.

[95] Maddern, W., Pascoe, G., Linegar, C., & Newman, P. (2017). 1 year, 1000 km: The oxford robotcar dataset. *The International Journal of Robotics Research*, 36(1), 3–15.

[96] Marcos, D., Volpi, M., Komodakis, N., & Tuia, D. (2017). Rotation equivariant vector field networks. In *ICCV* (pp. 5048–5057).

[97] Mejjati, Y. A., Richardt, C., Tompkin, J., Cosker, D., & Kim, K. I. (2018). Unsupervised attention-guided image-to-image translation. In *Advances in Neural Information Processing Systems* (pp. 3693–3703).

[98] Melekhov, I., Ylioinas, J., Kannala, J., & Rahtu, E. (2017). Relative camera pose estimation using convolutional neural networks. In *International Conference on Advanced Concepts for Intelligent Vision Systems* (pp. 675–687).: Springer.

[99] Mur-Artal, R., Montiel, J. M. M., & Tardos, J. D. (2015). Orb-slam: a versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5), 1147–1163.

[100] Newcombe, R. A., Lovegrove, S. J., & Davison, A. J. (2011). Dtam: Dense tracking and mapping in real-time. In *Proceedings of IEEE International Conference on Computer Vision*.

[101] Omari, S., Bloesch, M., Gohl, P., & Siegwart, R. (2015). Dense visual-inertial navigation system for mobile robots. In *Proceedings of IEEE International Conference on Robotics and Automation*.

[102] Oquab, M., Bottou, L., Laptev, I., & Sivic, J. (2014). Learning and transferring mid-level image representations using convolutional neural networks. In *CVPR*.

[103] Pentina, A., Sharmanska, V., & Lampert, C. H. (2015). Curriculum learning of multiple tasks. In *CVPR*.

[104] Philbin, J., Chum, O., Isard, M., Sivic, J., & Zisserman, A. (2007). Object retrieval with large vocabularies and fast spatial matching. In *2007 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1–8).: IEEE.

[105] Qin, Y., Song, D., Chen, H., Cheng, W., Jiang, G., & Cottrell, G. (2017). A dual-stage attention-based recurrent neural network for time series prediction. *arXiv preprint arXiv:1704.02971*.

[106] Rochan, M. & Wang, Y. (2015). Weakly supervised localization of novel objects using appearance transfer. In *CVPR*.

[107] Sabour, S., Frosst, N., & Hinton, G. E. (2017). Dynamic routing between capsules. In *NIPS* (pp. 3859–3869).

[108] Sattler, T., Weyand, T., Leibe, B., & Kobbelt, L. (2012). Image retrieval for image-based localization revisited. In *BMVC*, volume 1 (pp.$\tilde{4}$).

[109] Scaramuzza, D., Achtelik, M. C., Doitsidis, L., Friedrich, F., Kosmatopoulos, E., Martinelli, A., Achtelik, M. W., Chli, M., Chatzichristofis, S., Kneip, L., et al. (2014). Vision-controlled micro flying robots: from system design to autonomous navigation and mapping in gps-denied environments. *IEEE Robotics & Automation Magazine*, 21(3), 26–40.

[110] Schmid, K. & Hirschmüller, H. (2013). Stereo vision and imu based real-time ego-motion and depth image computation on a handheld device. In *Proceedings of IEEE International Conference on Robotics and Automation*.

[111] Schonberger, J. L. & Frahm, J.-M. (2016). Structure-from-motion revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 4104–4113).

[112] Sharmanska, V., Quadrianto, N., & Lampert, C. H. (2013). Learning to rank using privileged information. In *ICCV*.

[113] Shen, S., Michael, N., & Kumar, V. (2015). Tightly-coupled monocular visual-inertial fusion for autonomous flight of rotorcraft mavs. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on* (pp. 5303–5310).: IEEE.

[114] Shi, M., Caesar, H., & Ferrari, V. (2017a). Weakly supervised object localization using things and stuff transfer. *arXiv preprint arXiv:1703.08000*.

[115] Shi, M. & Ferrari, V. (2016). Weakly supervised object localization using size estimates. In *ECCV*.

[116] Shi, Z., Siva, P., & Xiang, T. (2017b). Transfer learning by ranking for weakly supervised object annotation. *arXiv preprint arXiv:1705.00873*.

[117] Shotton, J., Glocker, B., Zach, C., Izadi, S., Criminisi, A., & Fitzgibbon, A. (2013). Scene coordinate regression forests for camera relocalization in rgb-d images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2930–2937).

[118] Sibley, G., Matthies, L., & Sukhatme, G. (2010). Sliding window filter with application to planetary landing. *Journal of Field Robotics*, 27(5), 587–608.

[119] Simonyan, K. & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

[120] Siva, P., Russell, C., & Xiang, T. (2012). In defence of negative mining for annotating weakly labelled data. In *ECCV*.

[121] Snell, J., Swersky, K., & Zemel, R. (2017). Prototypical networks for few-shot learning. In *NIPS* (pp. 4077–4087).

[122] Song, H. O., Girshick, R. B., Jegelka, S., Mairal, J., Harchaoui, Z., Darrell, T., et al. (2014a). On learning to localize objects with minimal supervision. In *ICML*.

[123] Song, H. O., Lee, Y. J., Jegelka, S., & Darrell, T. (2014b). Weakly-supervised discovery of visual pattern configurations. In *NIPS*.

[124] Stallkamp, J., Schlipsing, M., Salmen, J., & Igel, C. (2011). The German Traffic Sign Recognition Benchmark: A multi-class classification competition. In *IJCNN* (pp. 1453–1460).

[125] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1–9).

[126] Takahashi, R., Matsubara, T., & Uehara, K. (2017). Scale-invariant recognition by weight-shared cnns in parallel. In *ACML* (pp. 295–310).

[127] Tang, P., Wang, X., Bai, X., & Liu, W. (2017). Multiple instance detection network with online instance classifier refinement. In *CVPR*.

[128] Tarrio, J. J. & Pedre, S. (2015). Realtime edge-based visual odometry for a monocular camera. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15 (pp. 702–710). Washington, DC, USA: IEEE Computer Society.

[129] Tateno, K., Tombari, F., Laina, I., & Navab, N. (2017). Cnn-slam: Real-time dense monocular slam with learned depth prediction. *arXiv preprint arXiv:1704.03489*.

[130] Toshev, A. & Szegedy, C. (2014). Deeppose: Human pose estimation via deep neural networks. In *CVPR*.

[131] Tudor Ionescu, R., Alexe, B., Leordeanu, M., Popescu, M., Papadopoulos, D. P., & Ferrari, V. (2016). How hard can it be? estimating the difficulty of visual search in an image. In *CVPR*.

[132] Uijlings, J. R., Van De Sande, K. E., Gevers, T., & Smeulders, A. W. (2013). Selective search for object recognition. *IJCV*, 104(2).

[133] Usenko, V., Engel, J., Stückler, J., & Cremers, D. (2016). Direct visual-inertial odometry with stereo cameras. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on* (pp. 1885–1892).: IEEE.

[134] Valada, A., Radwan, N., & Burgard, W. (2018). Deep auxiliary learning for visual localization and odometry. In *2018 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 6939–6946).: IEEE.

[135] Walch, F., Hazirbas, C., Leal-Taixe, L., Sattler, T., Hilsenbeck, S., & Cremers, D. (2017). Image-based localization using lstms for structured feature correlation. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 627–637).

[136] Wang, C., Xu, D., Zhu, Y., Martín-Martín, R., Lu, C., Fei-Fei, L., & Savarese, S. (2019). Densefusion: 6d object pose estimation by iterative dense fusion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 3343–3352).

[137] Wang, G., Wang, X., Fan, B., & Pan, C. (2017a). Feature extraction by rotation-invariant matrix representation for object detection in aerial image. *IEEE Geoscience and Remote Sensing Letters*, 14(6), 851–855.

[138] Wang, J., Wei, Z., Zhang, T., & Zeng, W. (2016). Deeply-fused nets. *arXiv preprint arXiv:1605.07716*.

[139] Wang, S., Clark, R., Wen, H., & Trigoni, N. (2017b). Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks. In *2017 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 2043–2050).: IEEE.

[140] Weiler, M., Hamprecht, F. A., & Storath, M. (2018). Learning steerable filters for rotation equivariant cnns. In *CVPR* (pp. 849–858).

[141] Weiss, S., Achtelik, M. W., Lynen, S., Chli, M., & Siegwart, R. (2012). Real-time onboard visual-inertial state estimation and self-calibration of mavs in unknown environments.

In *Robotics and Automation (ICRA), 2012 IEEE International Conference on* (pp. 957–964).: IEEE.

[142] Wen, Y., Zhang, K., Li, Z., & Qiao, Y. (2016). A discriminative feature learning approach for deep face recognition. In *ECCV* (pp. 499–515).

[143] Workman, S., Zhai, M., & Jacobs, N. (2016). Horizon lines in the wild. *arXiv preprint arXiv:1604.02129*.

[144] Worrall, D. E., Garbin, S. J., Turmukhambetov, D., & Brostow, G. J. (2017). Harmonic networks: Deep translation and rotation equivariance. In *CVPR*, volume 2.

[145] Wu, J., Li, D., Yang, Y., Bajaj, C., & Ji, X. (2018). Dynamic filtering with large sampling field for convnets. In *ECCV* (pp. 185–200).

[146] Wu, J., Yu, Y., Huang, C., & Yu, K. (2015). Deep multiple instance learning for image classification and auto-annotation. In *CVPR*.

[147] Xiang, Y., Schmidt, T., Narayanan, V., & Fox, D. (2017). Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *arXiv preprint arXiv:1711.00199*.

[148] Xin Wang, D. W., Mingcai Zhou, R. L., & Zha., G. (2016). Edge enhanced direct visual odometry. In *The British Machine Vision Association and Society for Pattern Recognition*.

[149] Xu, W. & Choi, D. (2016). Direct visual-inertial odometry and mapping for unmanned vehicle. In *International Symposium on Visual Computing* (pp. 595–604).: Springer.

[150] Xu, W., Choi, D., & Wang, G. (2018). Direct visual-inertial odometry with semi-dense mapping. *Computers & Electrical Engineering*, 67, 761–775.

[151] Xu, W., Keshmiri, S., & Wang, G. (2019a). Toward learning a unified many-to-many mapping for diverse image translation. *Pattern Recognition*, 93, 570 – 580.

[152] Xu, W., Shawn, K., & Wang, G. (2019b). Adversarially approximated autoencoder for image generation and manipulation. *IEEE Transactions on Multimedia*.

[153] Xu, W., Shawn, K., & Wang, G. (2019c). Stacked wasserstein autoencoder. *Neurocomputing*, 363, 195 – 204.

[154] Xu, W., Wu, Y., Ma, W., & Wang, G. (2019d). Adaptively denoising proposal collection for weakly supervised object localization. *Neural Processing Letters*, (pp. 1–14).

[155] Xu, X., Wang, G., Sullivan, A., & Zhang, Z. (2019e). Towards learning affine-invariant representations via data-efficient cnns. *arXiv preprint arXiv:1909.00114*.

[156] Xu, Y., Xiao, T., Zhang, J., Yang, K., & Zhang, Z. (2014). Scale-invariant convolutional neural networks. *arXiv preprint arXiv:1411.6369*.

[157] Yang, J., Jiang, Y.-G., Hauptmann, A. G., & Ngo, C.-W. (2007). Evaluating bag-of-visual-words representations in scene classification. In *Proceedings of the international workshop on Workshop on multimedia information retrieval* (pp. 197–206).: ACM.

[158] You, Q., Jin, H., Wang, Z., Fang, C., & Luo, J. (2016). Image captioning with semantic attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4651–4659).

[159] Yu, F. & Koltun, V. (2015). Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*.

[160] Yu, J., Hong, C., Rui, Y., & Tao, D. (2017). Multitask autoencoder model for recovering human poses. *IEEE Transactions on Industrial Electronics*, 65(6), 5060–5068.

[161] Yu, Z., Yu, J., Xiang, C., Fan, J., & Tao, D. (2018). Beyond bilinear: Generalized multimodal factorized high-order pooling for visual question answering. *IEEE transactions on neural networks and learning systems*, (99), 1–13.

[162] Zagoruyko, S. & Komodakis, N. (2016). Wide residual networks. *arXiv preprint arXiv:1605.07146*.

[163] Zeiler, M. D. & Fergus, R. (2014). Visualizing and understanding convolutional networks. In *ECCV*.

[164] Zhang, C., Li, H., Wang, X., & Yang, X. (2015a). Cross-scene crowd counting via deep convolutional neural networks. In *CVPR*.

[165] Zhang, J. m., Sclaroff, S., Lin, Z., Shen, X. H., Price, B., & Mech, R. (2016). Unconstrained salient object detection via proposal subset optimization. In *CVPR*.

[166] Zhang, Q., Nian Wu, Y., & Zhu, S.-C. (2018). Interpretable convolutional neural networks. In *CVPR* (pp. 8827–8836).

[167] Zhang, T., Lin, G., Cai, J., Shen, T., Shen, C., & Kot, A. C. (2019). Decoupled spatial neural attention for weakly supervised semantic segmentation. *IEEE Transactions on Multimedia*.

[168] Zhang, T., Qi, G.-J., Xiao, B., & Wang, J. (2017a). Interleaved group convolutions. In *CVPR*.

[169] Zhang, X., Liu, L., Xie, Y., Chen, J., Wu, L., & Pietikäinen, M. (2017b). Rotation invariant local binary convolution neural networks. In *ICCV Workshops* (pp. 1210–1219).

[170] Zhang, Y., Sohn, K., Villegas, R., Pan, G., & Lee, H. (2015b). Improving object detection with deep convolutional networks via bayesian optimization and structured prediction. In *CVPR*.

[171] Zhang, Z. & Brand, M. (2017). Convergent block coordinate descent for training tikhonov regularized deep neural networks. In *NIPS* (pp. 1721–1730).

[172] Zhao, C., Sun, L., & Stolkin, R. (2017). A fully end-to-end deep learning approach for real-time simultaneous 3d reconstruction and material recognition. *arXiv preprint arXiv:1703.04699*.

[173] Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., & Torralba, A. (2016). Learning deep features for discriminative localization. In *CVPR*.

[174] Zhou, G., Zhao, Y., Guo, F., & Xu, W. (2014). A smart high accuracy silicon piezoresistive pressure sensor temperature compensation system. *Sensors*, 14(7), 12174–12190.

[175] Zhou, Y., Ye, Q., Qiu, Q., & Jiao, J. (2017). Oriented response networks. In *CVPR* (pp. 4961–4970).

[176] Zhu, Y., Urtasun, R., Salakhutdinov, R., & Fidler, S. (2015). segdeepm: Exploiting segmentation and context in deep neural networks for object detection. In *CVPR*.

[177] Zitnick, C. L. & Dollár, P. (2014). Edge boxes: Locating object proposals from edges. In *ECCV*.

Appendix A

Proof and Optimization for Chapter 3

A.1 Jacobians of the IMU residual with respect to the IMU parameters

A.1.1 Background

We take the pre-integrated measurements Δp_G^k Δv_G^k ΔR_G^k as:

$$\Delta p_G^k = \frac{1}{2}\sum_{i=k}^{k+1}\{2(N-i)+1\}R_i^k(\hat{a}_i^i + b_a^i - R_G^i g^G)dt^2 + N v_k^G dt$$
$$\Delta v_G^k = \sum_{i=k}^{k+1} R_i^k(\hat{a}_i^i + b_a^i - R_G^i g^G)dt$$
$$\Delta R_G^k = \prod_{i=k}^{k+1} exp(\hat{\omega}_i^i + b_g^i)dt$$

The infinitesimal increment in so(3) with right hand-multiplication (38; 21) is

$$exp([\theta+\delta\theta]_\times) = exp([\theta]_\times)exp([J_r(\theta)^{-1}\delta\theta]_\times) \tag{A.1}$$

We take the same fist-order approximation for the logarithm introduced in (38)

$$log(exp([\theta]_\times exp([\delta\theta]_\times)))^\vee = \theta + J_r(\theta)^{-1}\delta\theta \tag{A.2}$$

where $J_r(\ )$ is the $SO(3)$ Jacobian.

Another efficient relation for linearization is from the adjoint representation

$$exp([\theta]_\times)R = Rexp([R^T\theta]_\times) \tag{A.3}$$

A.1.2 Jacobians

To calculate the Jacobian of the rotation error $r_{\Delta R_k^G}$ wrt the angular velocity bias b_g^i, we first move all of the increment terms to the right side of the equation according to the three equations in A.1.1:

$$\begin{aligned} r_{\Delta R_k^G}(b_g^i + \delta b_g^i) &= log((\hat{R}_{k+1}^k)^T R_G^k R_{k+1}^G)^\vee \\ &= log((\prod_{i=k}^{k+1} exp(\hat{\omega}_i^i + b_g^i + \delta b_g^i)dt)^T (R_k^G)^T R_{k+1}^G)^\vee \end{aligned}$$

Similarly to the other Jacobians, we apply the equations in A.1.1 to move all the increment terms to the right side and obtain:

$$\begin{aligned} \frac{\partial r_{\Delta p_G^k}}{\partial \delta b_a^k} &= -\frac{1}{2}\sum_{i=k}^{k+1}\{2(N-i)+1\}R_i^k dt^2 \\ \frac{\partial r_{\Delta v_G^k}}{\partial \delta b_a^k} &= -\sum_{i=k}^{k+1} R_i^k dt \\ \frac{\partial r_{\Delta R_G^k}}{\partial \delta b_a^k} &= 0 \\ \frac{\partial r_{\Delta p_G^k}}{\partial \delta b_g^k} &= \frac{1}{2}\sum_{i=k}^{k+1}\{C[\hat{a}_i^i + b_a^i]_\times B dt^2\} \\ \frac{\partial r_{\Delta v_G^k}}{\partial \delta b_g^k} &= \sum_{i=k}^{k+1} D[\hat{a}_i^i + b_a^i]_\times B dt \\ \frac{\partial r_{\Delta R_G^k}}{\partial \delta b_g^k} &= -J_r(r_{\Delta R_G^k})^{-1} exp([r_{\Delta R_G^k}]_\times)^T \sum_{i=k}^{k+1}\{[\prod_{m=i+1}^{k+1} exp\left([\hat{\omega}_i^i + b_g^i]_\times dt\right)]^T J_r\left((\hat{\omega}_i^i + b_g^i)dt\right)dt\} \end{aligned}$$

$$C = [\prod_{m=k}^{i-1}\{2(N-i)+1\}exp\left([\hat{\omega}_m^m + b_g^m]_\times dt\right)]$$

$$B = \sum_{l=k}^{i-1}\{[\prod_{m=l+1}^{i-1} exp\left([\hat{\omega}_m^m + b_g^m]_\times dt\right)]^T J_r\left((\hat{\omega}_l^l + b_g^l)dt\right)dt\}$$

$$D = \prod_{m=i+1}^{k+1} exp\left([\hat{\omega}_m^m + b_g^m]_\times dt\right)$$