ABSTRACT

Title of thesis: SalientDSO: Bringing Attention to
Direct Sparse Odometry

Huai-Jen Liang, Master of Science, 2018

Thesis directed by: Professor Yiannis Aloimonos
Department of Computer Science

Although cluttered indoor scenes have a lot of useful high-level semantic information which can be used for mapping and localization, most Visual Odometry (VO) algorithms rely on the usage of geometric features such as points, lines and planes. Lately, driven by this idea, the joint optimization of semantic labels and obtaining odometry has gained popularity in the robotics community. The joint optimization is good for accurate results but is generally very slow. At the same time, in the vision community, direct and sparse approaches for VO have stricken the right balance between speed and accuracy.

We merge the successes of these two communities and present a way to incorporate semantic information in the form of visual saliency to Direct Sparse Odometry – a highly successful direct sparse VO algorithm. We also present a framework to filter the visual saliency based on scene parsing. Our framework, *SalientDSO*, relies on the widely successful deep learning based approaches for visual saliency and scene parsing which drives the feature selection for obtaining highly-accurate and robust VO even in the presence of as few as 40 point features per frame. We provide exten-

sive quantitative evaluation of SalientDSO on the ICL-NUIM and TUM monoVO datasets and show that we outperform DSO and ORB-SLAM – two very popular state-of-the-art approaches in the literature. We also collect and publicly release a CVL-UMD dataset which contains two indoor cluttered sequences on which we show qualitative evaluations. To our knowledge this is the first framework to use visual saliency and scene parsing to drive the feature selection in direct VO.

SalientDSO: Bringing Attention to Direct Sparse Odometry

by

Huai-Jen Liang

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Master of Science
2018

Advisory Committee:
Professor Yiannis Aloimonos, Chair/Advisor
Dr. Cornelia Fermüller, Co-Advisor
Professor Behtash Babadi

Acknowledgments

I owe my gratitude to all the people who have made this thesis possible and because of whom my graduate experience has been one that I will cherish forever.

First and foremost I'd like to thank my advisor, Professor Yiannis Aloimonos for giving me an invaluable opportunity to work on challenging and extremely interesting projects over the past two years. He has always motivated me and encouraged me to work on a challenging and interesting project. It has been a pleasure to work with and learn from such an extraordinary individual.

Next, I would also like to thank my co-advisor, Dr. Cornelia Fermüller. Without her continuing and unfailing support, this thesis would have been a distant dream. Thanks are due to Professor Behtash Babadi for accepting the invitation to serve on my thesis committee and for sparing his invaluable time reviewing the manuscript.

My colleagues at the Computer Vision laboratory have enriched my graduate life in many ways and deserve a special mention. Thanks Nitin J. Sanket for supporting my work on this thesis as well. His insightful suggestion and comments inspired and influenced the idea of the presented method.

I owe my deepest thanks to my family - my mother and father who have always stood by me and guided me through my career, and have pulled me through against impossible odds at times. Words cannot express the gratitude I owe them.

It is impossible to remember all, and I apologize to those I've inadvertently left out.

Lastly, thank you all!

Table of Contents

List of Tables

List of Figures

List of Abbreviations

DSO	Direct Sparse Odometry
GAN	Generative Adversarial Network
PSPNet	Pyramid Scene Parsing Network
SalientDSO	Salient Direct Sparse Odometry
SLAM	Simultaneously Localization and Mapping
VO	Visual Odometry

Chapter 1: Introduction

Simultaneous Localization and Mapping (SLAM) and Visual Odometry (VO) algorithms have taken center stage in the recent years due to their wide-spread usage. They play a prominent part in the perception and planning pipelines of self-driving cars, autonomous quadrotors, augmented and virtual reality. The never ending quest to come up with realtime solutions for these methods whilst being as accurate as their offline counterparts has led to alternative problem formulations in terms of constraints and optimization methods [1–4].

Not so long ago, the field was dominated by indirect methods [2, 5, 6] which rely on feature matching and foundations of multi-view geometry coupled with windowed optimization to build a map of the scene and obtain accurate poses. 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. Some works have improved upon the previous approaches in-terms of speed and accuracy by incorporating prior knowledge such as the dynamics of the system and/or data from more sensors such as inertial measurement units [7], time-of-flight sensors [8] etc. However, minimalism is a trend forward, i.e., trying to achieve the same tasks with a minimal number of sensors. In the scope of this thesis, we focus on a monocular

VO solution. The current state-of-the-art in monocular approaches which have the best compromise of speed and accuracy are direct sparse approaches such as Direct Sparse Odometry (DSO) [9].

However, object centric SLAM approaches are more robust by nature due to the high level semantics used in the formulation. Lately, joint optimization of 3D poses, stucture and labelled object locations has improved the state-of-the-art significantly. These frameworks rely on the widely successful deep learning based object recognition engine and pose graph optimization frameworks, combining both low-level geometric features and the high-level semantics.

However, humans perform the task of mapping very differently. The human visual system interprets the scene for various tasks like recognition, segmentation, tracking and navigation by making a series of fixations [10]. This is called the Active approach [11–13], whilst the traditional approach is called the Passive approach (See Table 1.1). These fixations lie in the proto-segmentation of the salient objects/locations in the scene. The word proto-segmentation refers to the fact that a segmentation around the fixation point may lead to partial/complete segmentation of an object, which depends on the scenario. Solving the problem of recognition and tracking along with segmentation is like a chicken-egg problem. One would need a good segmentation for recognition and tracking and vice-versa. An Expectation-Maximization (EM) type of scheme, where one would jointly/alternatively optimize for the segmentation and recognition/tracking has gained popularity in literature lately, due to the advancement of fast and accurate optimization frameworks.

Very recently, this philosophy of fixation and attention has started to gain

Table 1.1: Active vs Passive approach for computer vision tasks.

Task	Passive approach	Active approach
Segmentation	Graph cut or super-pixel based methods.	Fixation based region segmentation and recognition in a feedback loop.
Recognition	Sliding window of filter banks with a classification algorithm for final prediction.	Saliency/fixation based segmentation/clustering followed by selection of attributes and sliding window of filters with a simple classification algorithm.
Tracking and Failure recovery	Making an online dictionary for robustness against changes and use detection for failure recovery.	Tightly couple saliency into the tracking filter to reduce search space and use salient regions for failure recovery. By doing so, we introduce high level semantics into the low level processes (feedback).
Navigation and Mapping	Map based on features based on image gradients.	Map only using salient region features or objects obtained using fixation based segmentation. Take advantage of the semantic relationships between differently labeled regions.

popularity in the robot navigation community [14–17]. This is based on the fact that humans perform the task of mapping very differently from how it has been done in the robotics literature. They build "sematic/toplogical" maps to traverse the scene. This thesis combines the concepts used by humans and robotics literature to present a framework of indoor visual odometry in which the features are selected based on a visual saliency map that is obtained by human eye tracking data. This work aims to mimic the qualitative human vision in the framework of direct VO.

1.1 Contribution

The key contributions of this thesis are:

- We present a framework of indoor visual odometry in which the features are selected based on a visual saliency map (Sample output is shown in Fig. 1.1).
- We present a method to filter saliency map based on scene parsing.
- We provide experimental results on various simulated and real indoor environments to demonstrate the improved performance of the proposed approach with comparisons to the state-of-the-art.

1.2 Outline

The rest of the thesis is organized as follows: Chapter. 2 presents the pipeline of the proposed SalientDSO framework. In Chapter. 3, we introduces the required preliminaries and adopt VO backbone algorithm DSO [9]. Chpater. 4 describes the deep network used to predict saliency. Chapter. 5 presents the deep neural model for retrieving semantic information. Chapter. 6 describes the visual saliency and scene parsing driven point selection algorithm used in SalientDSO. Detailed experiments along with quantitative and qualitative results are given in Chapter. 7. We finally conclude the thesis in Chapter. 8 with parting thoughts on future work.

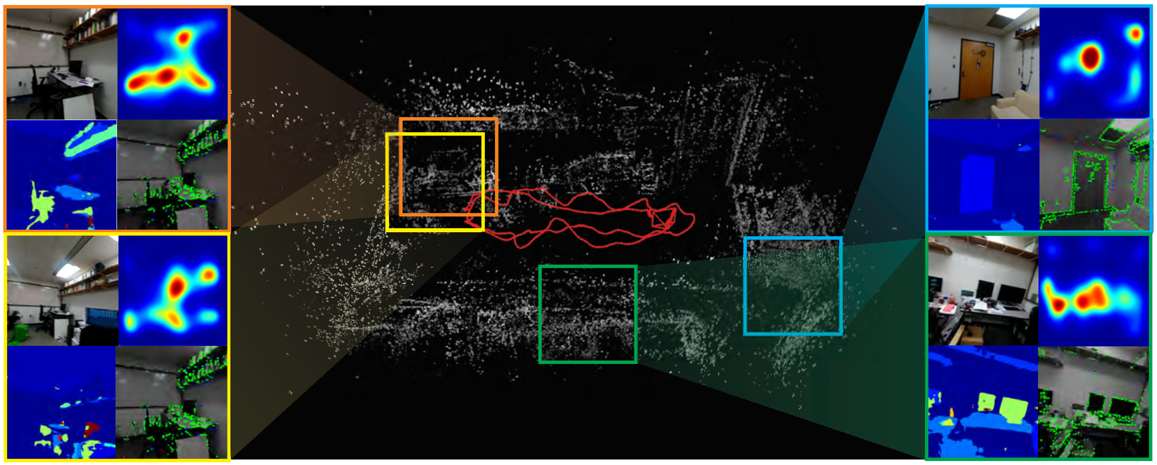

Figure 1.1: Sample point-cloud output of SalientDSO which does not have loop closure or global bundle adjustment. The insets show the corresponding image, saliency, scene parsing outputs and active features. Observe that features from non-informative regions are almost removed approaching object centric odometry.

Chapter 2: SalientDSO

The overview of SalientDSO is presented in Fig. 2.1. The blue parts of the Fig. 2.1 show our contribution which constitutes the pre-processing step. The SalientDSO contains following components:

- **DSO** serves as the Visual Odometry backbone
- **SalGAN** predicts saliency map of a given image
- **Scene Parsing** retrieves semantic information of a given image
- **Features/Points Selection** uses semantic information to filter saliency map and select features/points according to the filtered saliency map

Each components will be detailed in the following chapters. In brief, SalientDSO extracts information from interesting regions/objects in observed environment. Gathering this information, SalientDSO estimates camera pose as well as 3D world model simultaneously by tracking salient features/points and optimizing estimation with Gaussian-Newton algorithm in a sliding window manner. By using salient information in a scene, SalientDSO performs better in accuray and much robust in a severe parameter setting compare to the state-of-the-art algorithms.

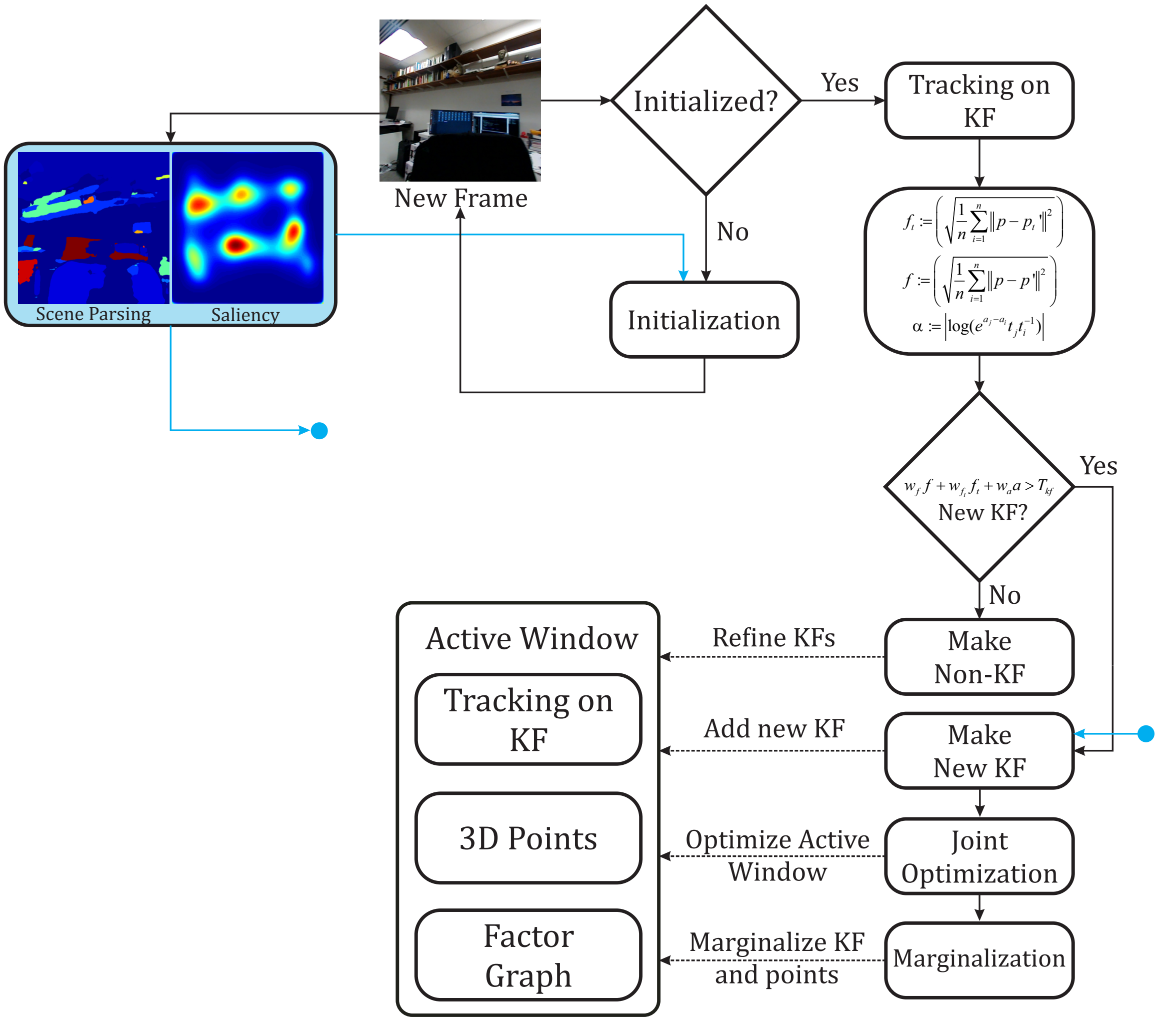
New Frame
Scene Parsing
Saliency
Initialized?
Yes
No
Initialization
Tracking on KF
$f_t := \left(\sqrt{\frac{1}{n}\sum_{i=1}^{n}\|p - p_t'\|^2}\right)$
$f := \left(\sqrt{\frac{1}{n}\sum_{i=1}^{n}\|p - p'\|^2}\right)$
$\alpha := \left|\log(e^{a_j - a_i} t_j t_i^{-1})\right|$
$w_f f + w_{f_t} f_t + w_a a > T_{kf}$
New KF?
Yes
No
Make Non-KF
Make New KF
Joint Optimization
Marginalization
Refine KFs
Add new KF
Optimize Active Window
Marginalize KF and points
Active Window
Tracking on KF
3D Points
Factor Graph

Figure 2.1: Algorithmic overview of SalientDSO, blue parts show our contributions.

Chapter 3: Monocular SLAM and Visual Odometry

Simultaneous localization and mapping is a process of estimating the state of a robot using on-board sensors, such as cameras, IMU units, and GPS. Simultaneous localization and mapping is a key problem in computer vision as self driving autonomous becomes much more popular in recent years.

In this thesis, the Simultaneous localization and mapping serves as the backbone of the whole system. Instead of evenly extracting features from an image, we concentrate in regions which are interesting in an environment. In the next section, a powerful techniques presented in [9] adopted here will be analyzed in detail.

3.1 Introduction to Direct Sparse Odometry(DSO)

While for a long time, Simultaneous localization and mapping was dominated by feature-based (indirect) method [2,5,18], more and more methods, such as direct [19] and dense [3,4,19–22], have emerged in recent years.

3.1.1 Different Formulations

Despite there are different formulation, underlying all is a probabilistic model which estimates unknown X (3D world model and camera motion) based on noise measurements Y (images). Typically, a Maximum Likelihood approach is applied.

$$X = argmax_X P(Y \mid X) \tag{3.1}$$

According to the description in [9], different formulations can be described as following:

3.1.1.1 Direct vs. Indirect

Indirect methods will first pre-process raw sensor measurement to generate intermediate representation, such as SIFT [23], SURF [24], and ORB [5]. Second, as soon as keypoints have been extracted and matched across different views, they are feeded into the underlying probabilistic model as measurement Y to estimate world model and camera motion.

Direct methods directly use the raw sensor measurement (Intensity values) as noise measurement Y, instead of generating intermediate representation.

3.1.1.2 Dense Vs. Sparse

Dense methods gather information from and reconstruct all pixels in an image, while sparse methods [19] only utilize a selected set (corners, edges).

More importantly, their geometric prior are different. Dense methods can establish connectedness between neighbor pixels and formulate as geometric prior while sparse methods can't. Such prior is necessary for reconstructing a dense world model [3, 21, 25].

3.1.2 Implementation Details

DSO introduced a direct and sparse method. The main benefits of using keypoints as in indirect method is their ability to provide robustness to photometric and geometric distortions present in an image. However, with a more precise sensor model, auto-exposure and gamma correction are not unknown noise. It benefits direct approaches since direct approaches process image information down to pixel intensities and can be more informative.

Another benefits of DSO is that because of introducing geometry prior, optimization of dense methods in real time is infeasible. However, sparse methods can be solved efficiently by Schur complement since its Hessian structure is diagonal.

DSO contains two parts, front end for frames/points selection and initialization and back end for optimization. The whole pipeline is shown in Fig. 2.1 colored in black. Note that in the proposed framework, points selection is replaced with our proposed method in Chapter.6.

3.1.2.1 Calibration

In addition to geometric camera model, it is necessary to do photometric camera calibration in direct method. Following the formation in [26], a non-linear response function $G : \mathbb{R} \to [0, 255]$ with lens attenuation $V : \Omega \to [0, 1]$ maps irradiance B_i to the respective intensity value I_i. This is given by

$$I_i(x) = G\left(t_i V(x) B_i(x)\right) \tag{3.2}$$

where t_i is the exposure time. To get a photometrically corrected pixel value,

$$I_i'(x) = t_i Bi(x) = \frac{G^{-1}\left(I_i(x)\right)}{V(x)} \tag{3.3}$$

is applied to each video frame as very first step. Note that, in the remainder of this thesis, I_i will always refer to the photometrically corrected image I_i'.

3.1.2.2 Front end

The front end is the part of algorithm that handles the following:

- **Initial Frame Tracking:** A new frame is tracked with respect to the latest keyframe by using conventional two-frame direct image alignment, a multiscale image pyramid and a constant motion model. If tracking is fail, DSO attempt to recover a motion by trying 27 different small rotations.

- **Keyframe Creation:** Similar to ORB-SLAM [5], DSO take as many keyframes as possible, and then sparsify afterwards by marginalizing redundant keyframes. There are three rules to decide if a new keyframe is needed:

  - Mean square optical flow $f = \left(\frac{1}{n}\sum_{i=1}^{n} \|p - p'\|^2\right)^{\frac{1}{2}}$
  - Mean flow without rotation $f_t = \left(\frac{1}{n}\sum_{i=1}^{n} \|p - p_t'\|^2\right)^{\frac{1}{2}}$, where p_t' is the warped point position with identity rotation matrix.
  - Relative brightness factor $a = \left|log\left(e^{a_j - ai} t_j t_i^{-1}\right)\right|$

  After all, DSO combines all three and create a new key frame if $w_f f + w_{f_t} f_t + w_a a > T_{kf}$. Here the symbols have the same meaning as in [9].

- **Candidate point tracking:** Candidate points are selected using the approach described in Sec.6. These points are then tracked by using discrete search along epipolar line and minimizing the photometric error E_{photo} given by Eq. 3.6. The computed depth and variance is used to constrain the search interval for subsequent frame as described in LSD-SLAM [4].

- **Outlier rejection and occlusion detection:** Point observations which have a E_{photo} above a certain threshold are removed as outliers and excluded for further computation.

- **Parameters initialization:** This step provides the initial estimates of all parameters for optimizing the non convex error E_{photo}. The initial camera pose is computed from direct image alignment and the initial point's depth is from candidate point tracking.

- **Candidate point activation:** New candidates points replace the old marginalized points. The new points are chosen by projecting onto the current frame and maximizing the distance between projection of any existing active points.

- **Marginalization:** This step decides which points and frames should be marginalized. A KF will be marginalized if less than 5% of points are visible in the latest frame. If there are more than N_f (fixed at 7) KFs, a KF which is far from current frame and close to any other KFs will be marginalized.

3.1.2.3 Back end

The back end contains a factor graph which performs continuous windowed optimization using the approach by Leutenegger et al. [27]. It optimizes the total error (3.6) using Gaussian-Newton algorithm in a sliding window manner. The error functions are defined as the following:

For a single active point p, its photometric error on keyframe j is defined as

$$E_{pj} = \sum_{p \in N_p} w_p \left\| (I_j[p'] - b_j) - \frac{t_j e^{a_j}}{t_i e^{a_i}} (I_i([p] - b_i) \right\|_\gamma \tag{3.4}$$

where p' is the projection of point p on KF j, $\{t_i, t_j\}$ are the exposure time for images $\{I_i, I_j\}$, $\|\|_\gamma$ is the Huber norm, a_i, a_j, b_i, b_j are brightness transfer function parameters, N_p is the residual pattern with eight surrounding neighbors and gradient depending weights w_p is given by

$$w_p = \frac{c^2}{c^2 + \|\bigtriangledown I_i(p)\|_2^2} \tag{3.5}$$

The full photometric error over all active points and keyframes is defined as

$$E_{photo} = \sum_{i \in F} \sum_{p \in P_i} \sum_{j \in obs(p)} E_{pj} \tag{3.6}$$

where F indicates all active keyframes, P_i indicates all active points in keyframe i, $obs(p)$ indicates all frames' observation in which point p is visible.

When the active set of variables becomes too large, DSO follows [27] to marginalize points and frames.

Chapter 4: Saliency Prediction

Saliecny prediction is popular in research for many years. Saliency prediction is a task to estimate the probability of a region in an image that attracts human's attention. It's prediction can be used as guides for other computer vision tasks or user studies.

Similar to other computer vision tasks, researchers started with extracting information from low-level features. [28] extracts low-level features in multiple scales and combine them to form a saliency prediction. By combining graph model [29] and mid- and high-level features [30, 31], they achieved predicting better saliency prediction or eye fixations.

In recent years, more and more deep learning solutions [32–41] has been proposed and significantly improved performance. According to MIT saliency benchmark, nine out of top ten results are deep learning solution.

In this thesis, Saliency prediction serves as the guide for candidate points' selection. We not only consider high gradient pixels, but also select points from higher saliency region with higher probability. Candidate points' selection will be discussed in detail in Chapter. 6. In the next section, a powerful model presented in [42] adopted here will be analyzed in detail.

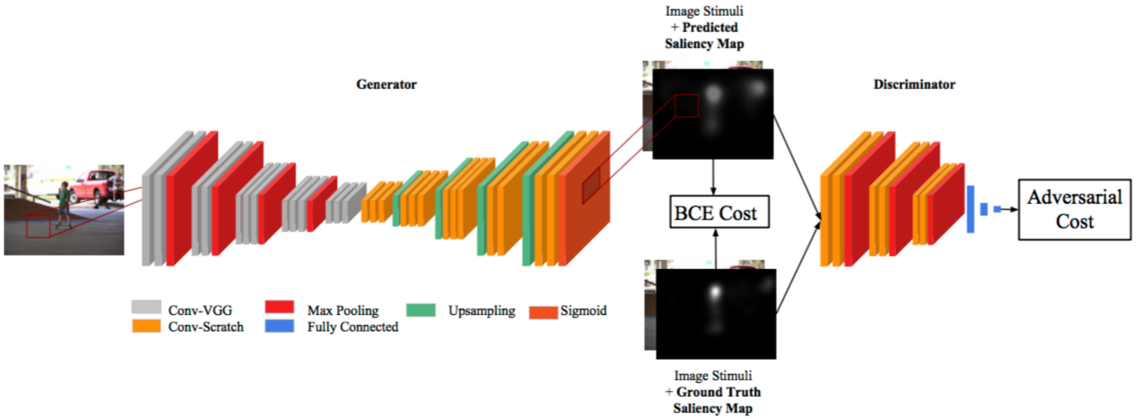

Figure 4.1: The overall architecture of SalGAN from [42].

4.1 Introduction to SalGAN

SalGAN [42] introduced the use of generative adversarial network (GANs) [43] for saliency prediction. It contains generator and discriminator. Generator is a deep convolutional neural network trained on adversarial loss (L_{GAN} in Eq. 4.2) which includes binary cross entropy loss (L_{BCE} in Eq. 4.1) to produce a downsampled saliency map and dscriminator is a shallower network as compared to the generator which is trained to solve binary classification between saliency map produced by generator and the groundtruth one. The overall architecture is shown in Fig. 4.1. SalGAN [42] is trained on SALICON [44] and evaluated on both SALICON [44] and MIT300 [45].

4.1.1 Generator

Generator is a encoder-decoder like network. The encoder part contains max pooling layers which decrease the size of feature maps. Encoder's structure is identical to VGG-16 [46] and its weights are initialized with the weights trained on the ImageNet dataset [47]. Decoder part is identical to encoder part with reversed ordering. Decoder's weights are randomly initialized.

The binary cross entropy loss between predicted saliency map $\hat{S}$ and groundtruth S is defined as

$$L_{BCE} = -\frac{1}{N}\sum_{j=1}^{N} S_j log(\hat{S}_j) + (1 - S_j) log(1 - \hat{S}_j) \tag{4.1}$$

where S_j is the probability of pixel I_j being fixated.

4.1.2 Discriminator

Discriminator, in short, is a network trained to distinguish between samples from the true distribution and generated samples. Its detail architecture is described in Table 4.1.

The final loss which includes content loss (4.1) for adversarial training is defined as

$$L_{GAN} = \alpha \cdot L_{BCE} - logD(I, \hat{S}) \tag{4.2}$$

where $D(I, \hat{S})$ is the probability of fooling the discriminator.

Some sample results are shown in Fig. 4.2. One can clearly notice that walls, floors, and ceilings have lower probability of being fixated on, which is the main idea

Table 4.1: Detail architecture of discriminator.

layer	depth	kernal	stride	pad	activation
conv1_1	3	1 x 1	1	1	ReLU
conv1_2	32	3 x 3	1	1	ReLU
pool1		2 x 2	2	0	
conv2_1	64	3 x 3	1	1	ReLU
conv2_2	64	3 x 3	1	1	ReLU
pool2		2 x 2	2	0	
conv3_1	64	3 x 3	1	1	ReLU
conv3_2	64	3 x 3	1	1	ReLU
pool3		2 x 2	2	0	
fc1	100				tanh
fc2	2				tanh
fc3	1				sigmoid

of the proposed framework.

4.2 SALICON Dataset

Saliency in Context(SALICON) [44] is a publicly available large dataset containing saliency annotated MSCOCO [48] images. This dataset contains 10000 training images, 5000 validation images, and 500 testing images. Some examples are shown in Fig. 4.3.

4.2.1 Data Collection

SALICON [44] proposed a novel approach to simulate the natural viewing behavior of humans. This allows one to collect the probability of visual attention by aggregating mouse trajectories from different users, instead of recording viewing behavior with eye-tracker.

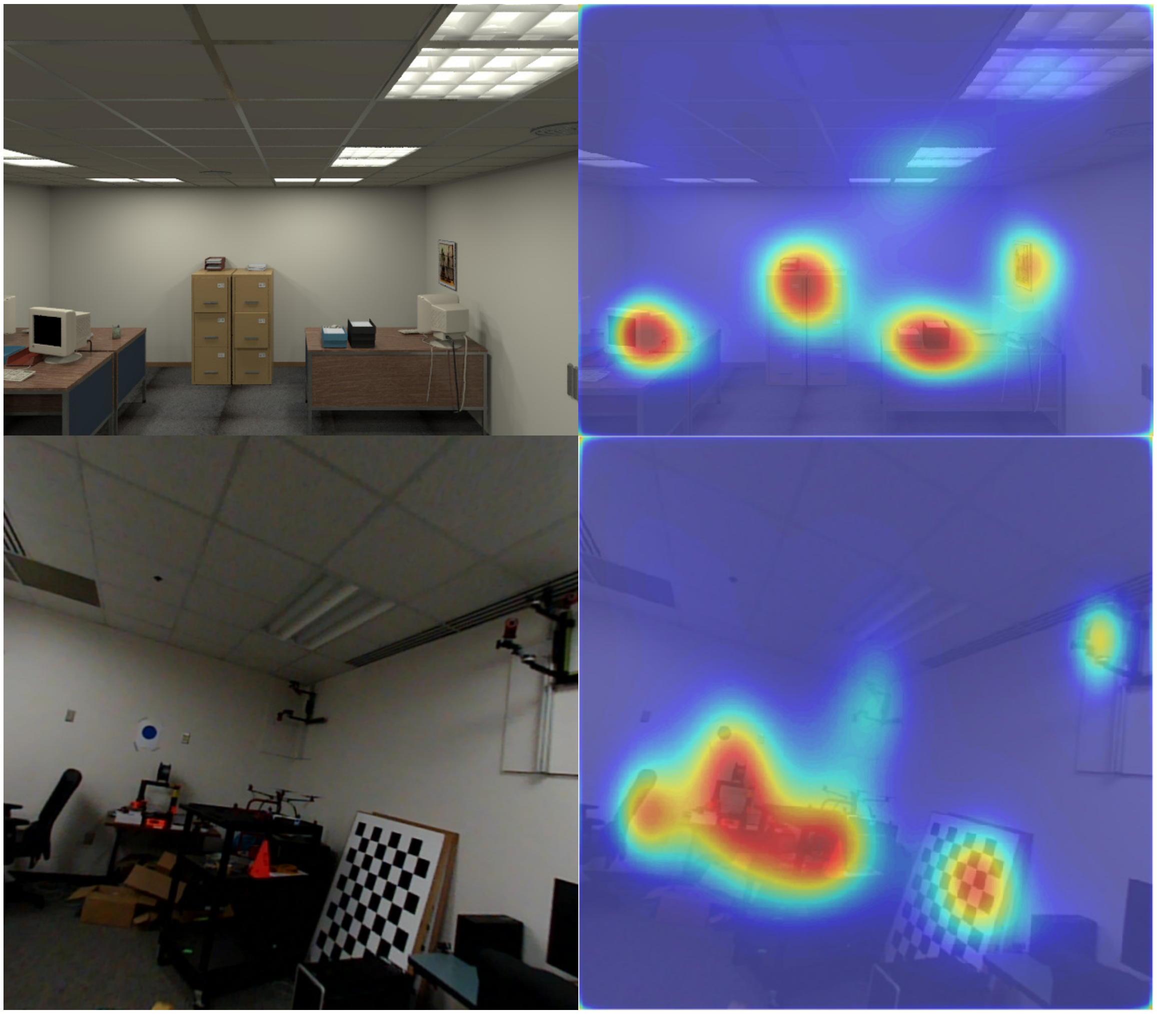

Figure 4.2: Left column: Input image, Right column: Saliency overlayed on input image.

4.2.2 Subjects

The experiment is deployed on the Amazon Mechanical Turk to enable large scale data collection.

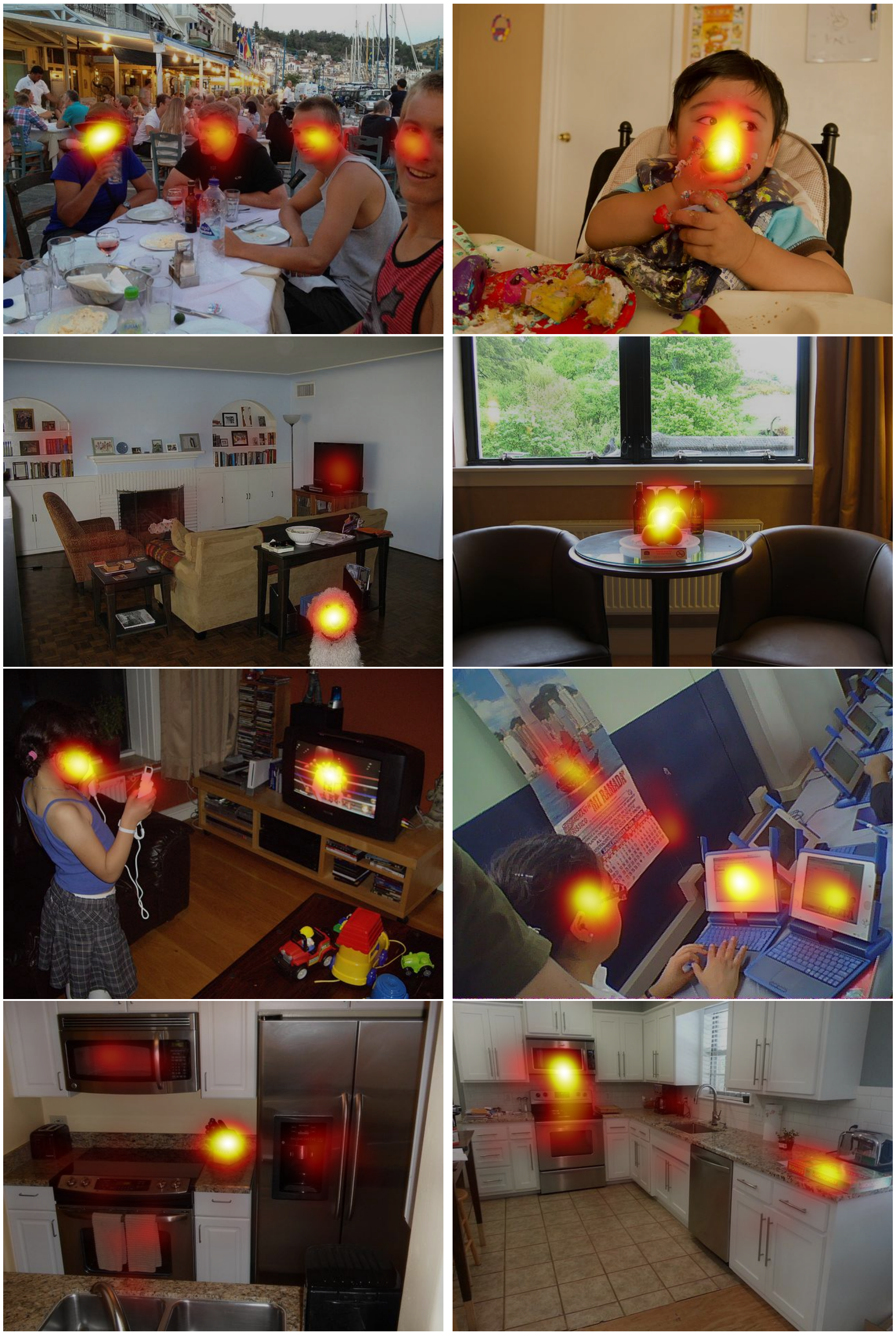

Figure 4.3: Examples of SALICON [44].

Chapter 5: Scene Parsing

The saliency produced by SalGAN is concentrated around a fixation point inside the object and is fuzzy. Moreover, the saliency map is not very robust to viewpoint and illumination changes as the fixation point does not remain constant. Therefore, we utilize semantic information to filter the saliency. In this chapter, we introduce the deep nerual network presented in [49] and the training data for our application.

5.1 Introduction to Pyramid Scene Parsing Network

To obtain semantic information from a scene, we adopt Pyramid Scene Parsing Network [49] for retrieving semantic labels of every pixel in an image. In brief, Pyramid Scene Parsing Network (PSPNet) is a deep neural network for pixel-level prediction tasks. PSPNet uses CNN layers to extract features, then a pyramid parsing module is applied to harvest different sub-region representation, followed by up-sampling and concatenation layers to form the final feature representation. The final features are then fed into more CNN layers to obtain a pixel-level prediction. The overall architecture is shown in Fig. 5.1. PSPNet is trained on ADE20K dataset [50], since ADE20K contains various indoor scenes and objects which is suitable for

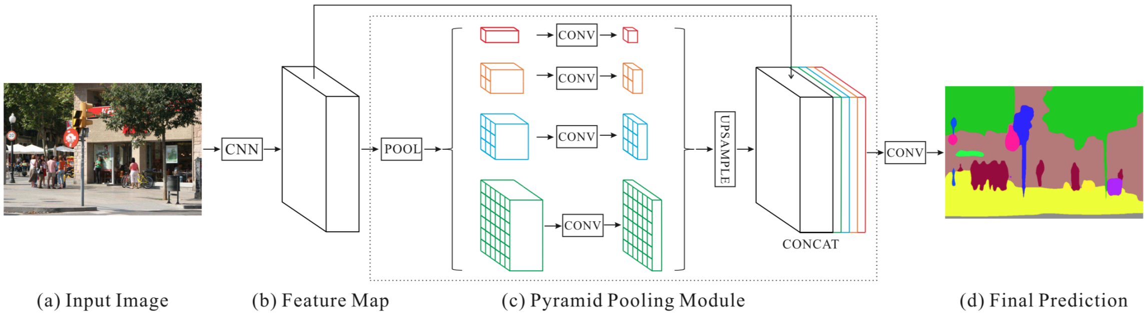

Figure 5.1: The overall architecture of PSPNet from [49].

our proposed framework.

5.1.1 Pyramid Pooling Module

In [51], it is shown that the empirical receptive field of CNN is much smaller than the theoretical one on high-level layers, which makes conventional networks not sufficiently incorporate the momentous global scenery prior. Moreover, motivated by some important observations from ADE20K dataset [50] and several common issues for complex-scene parsing, such as mismatched relationship, confusion categories, and inconspicuous classes, PSPNet introduce the pyramid pooling module, which empirically proves to be an effective global contextual prior.

As illustrated in part (c) of Fig. 5.1, pyramid pooling module fuses features under different pyramid scales. It first pools features of the previous layer with different kernel size and strides, which are determined by the desired output bin sizes. Then, 1×1 convolution layers are applied to reduce the dimension of context representation to $\frac{1}{N}$ of the original one if the level size of pyramid is N. At last, reduced representation is upsampled to the same size as the original feature map via bilinear interpolation and different levels of features are concatenated with the original feature to form the final pyramid pooling global features.

5.1.2 Network Architecture

As shown in Fig. 5.1, the whole network contains three parts: ResNet [52], pyramid pooling module, and final convolution layer. Given an input image, it is fed into a pretrained ResNet model with the dilated network strategy [53, 54] to extract the feature map with $\frac{1}{8}$ size of the input image (part (b) in Fig. 5.1). On top of the map, pyramid pooling module is applied to gather context information and representation from different levels are concatenate with the original feature (part(c) in Fig. 5.1). In PSPNet, 4-level pyramid with bin sizes of 1×1, 2×2, 3×3, and 6×6 is used. It is followed by the final convolution layer to from the final prediction map (part (d) in Fig. 5.1).

5.2 ADE20K Dataset

ADE20K [50] is a publicly available large dataset containing diverse annotations of scenes, objects, parts of objects, and in some cases even parts of parts. There are 20,210 images in the training set, 2,000 images in the validation set, and 3,000 images in the testing set. For scene parsing benchmark, it contains 150 object and stuff classes. Some examples are shown in Fig. 5.2.

5.2.1 Data Collection

Images come from the LabelMe [55], SUN datasets [56], and Places [57] and were selected to cover the 900 scene categories defined in the SUN database.

5.2.2 Subjects

Images were annotated by a single expert to achieve naming consistencies for open vocabulary naming.

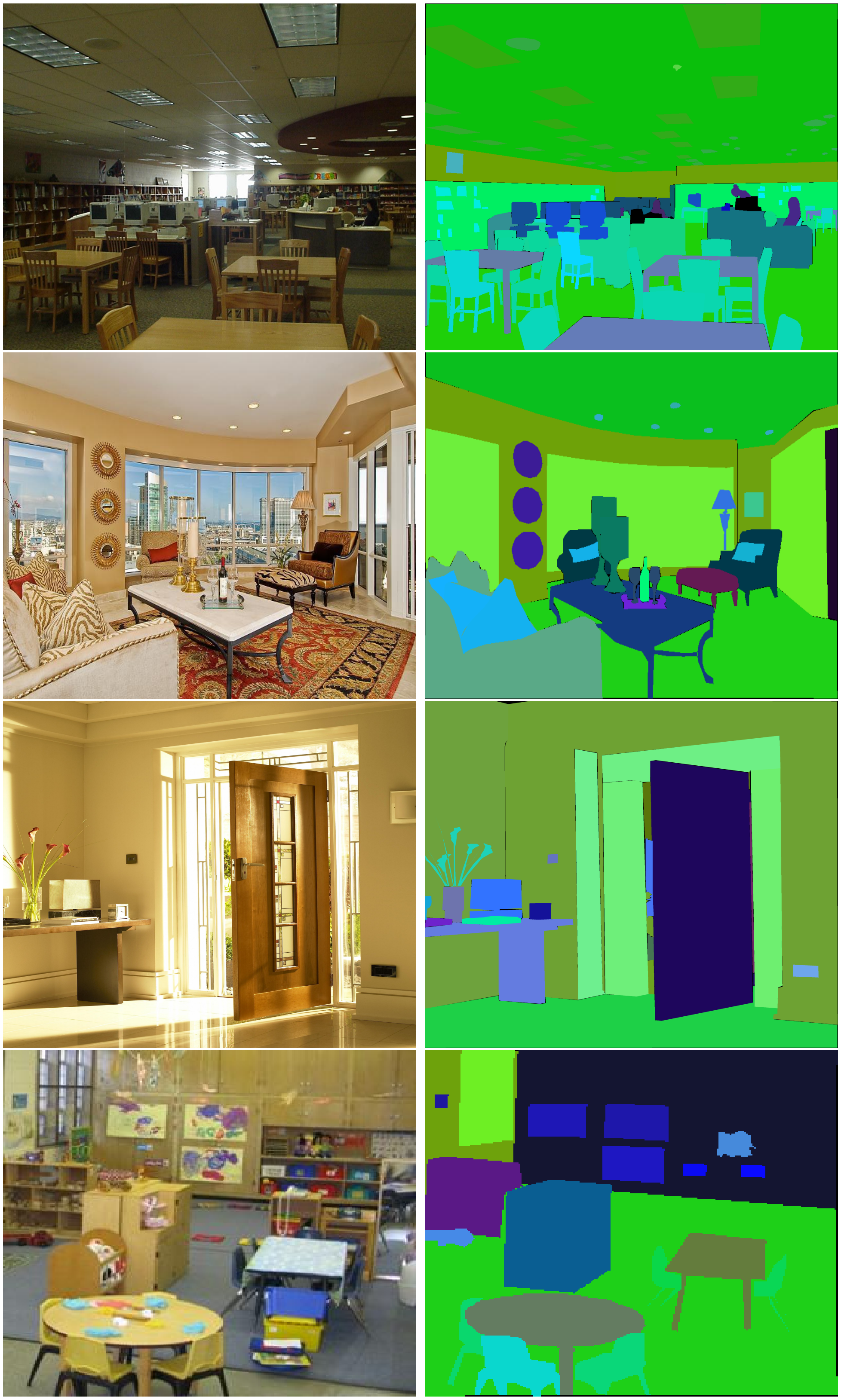

Figure 5.2: Examples of ADE20K [50]. Left: color images. Right: class label map.

Chapter 6: Candidate Point Selection

Instead of uniformly selecting candidate points from an image as in DSO, we select points based on saliency. This is very helpful where the scene has a lot of objects or in a clutter which can be found generally in indoor scenes.

6.1 Implementation Details

6.1.1 Saliency Prediction and Filtering

We feed input images into the SalGAN which is introduced in Chapter. 4 and generate an intermediate saliency map $\hat{S}$. As mentioned in Chapter. 5, the saliency produced by SalGAN is concentrated around a fixation point inside the object and is fuzzy. Moreover, the saliency map is not very robust to viewpoint and illumination changes as the fixation point does not remain constant. Therefore, we utilize semantic information to filter the saliency. The idea is to weigh down the saliency of uninformative regions, such as walls, ceilings and floors, and make saliency consistent across objects with the same semantic meanings.

Once the per-pixel semantic information C is obtained from PSPNet in Chap-

Algorithm 1: Saliency prediction and filtering.

```
Data: Input image $I$, Pre-defined weights $w_C$
Result: Predicted final saliency $\hat{S}^{\text{final}}$
$\hat{S} = \text{SalGAN}(I)$;
$C = \text{PSPNet}(I)$;
for $\forall \{x_j, y_j\} \in I$ do
    $\hat{S}_j^{\text{weighted}} = w_C(C_j)\hat{S}_j$;
end
for $\forall \{x_j, y_j\} \in I$ do
    $\hat{S}_j^{\text{final}} = \text{median}\left\{\hat{S}_i^{\text{weighted}}, \forall i \in C_j\right\}$;
end
```

ter. 5, the predicted saliency map $\hat{S}$ is filtered by:

$$\hat{S}_j^{\text{weighted}} = w_C(C_j)\hat{S}_j \tag{6.1}$$

Here, w_C are the predefined weights obtained empirically for different classes. To smooth and maintain a consistent saliency map for each class, each pixel is replaced by the median of saliency for its respective class:

$$\hat{S}_j^{\text{final}} = \text{median}\left\{\hat{S}_i^{\text{weighted}}, \forall i \in C_j\right\} \tag{6.2}$$

All steps to generate $\hat{S}^{\text{final}}$ are summarized in Algorithm 1.

6.1.2 Features/Points Selection

First, we split an image into $K \times K$ patches. For a patch M_i, we not only compute the median of gradient as a region-adaptive threshold, but also compute the median of saliency as a region-adaptive sampling weight sw_i. Therefore, for each patch, the sampling weight sw_i is computed as:

$$sw_i = \text{median}\left\{\hat{S}_j^{\text{final}}, \forall j \in M_i\right\} + s_{\text{smooth}} \tag{6.3}$$

where s_{smooth} is a laplacian smoothing term used to control the bias on a salient region and the probability of a patch M_i being sampled is:

$$\boldsymbol{P}_S(M_i) = \frac{sw_i}{\sum_{m \in M} sw_m} \tag{6.4}$$

Secondly, once a patch M_i has been selected, we further split M_i into $d \times d$ blocks. For each block, we select the pixel with the highest gradient only if it surpasses the region-adaptive threshold. With this strategy, we can select points which are well distributed in this salient region. In order to extract information from where no high-gradient pixels are present, we follow the same approach as DSO and run two more passes to select pixels with weaker gradient in a larger sub-region with a lower gradient threshold and an increased d. A summary of the whole selection method is given in Algorithm 2.

Fig. 6.1 shows the selected points for some example scenes. We compare our selection based on saliency to the uniform selection adopted by DSO. One can easily notice that texture-less and mostly identical parts, such as walls, floors and ceilings, are down weighted in our pipeline. As demonstrated in Section 7, this helps us trade the weak features on the floors and ceilings for weak features on objects where the saliency is generally higher - thus, in-turn, making the feature selection more robust and object-centric.

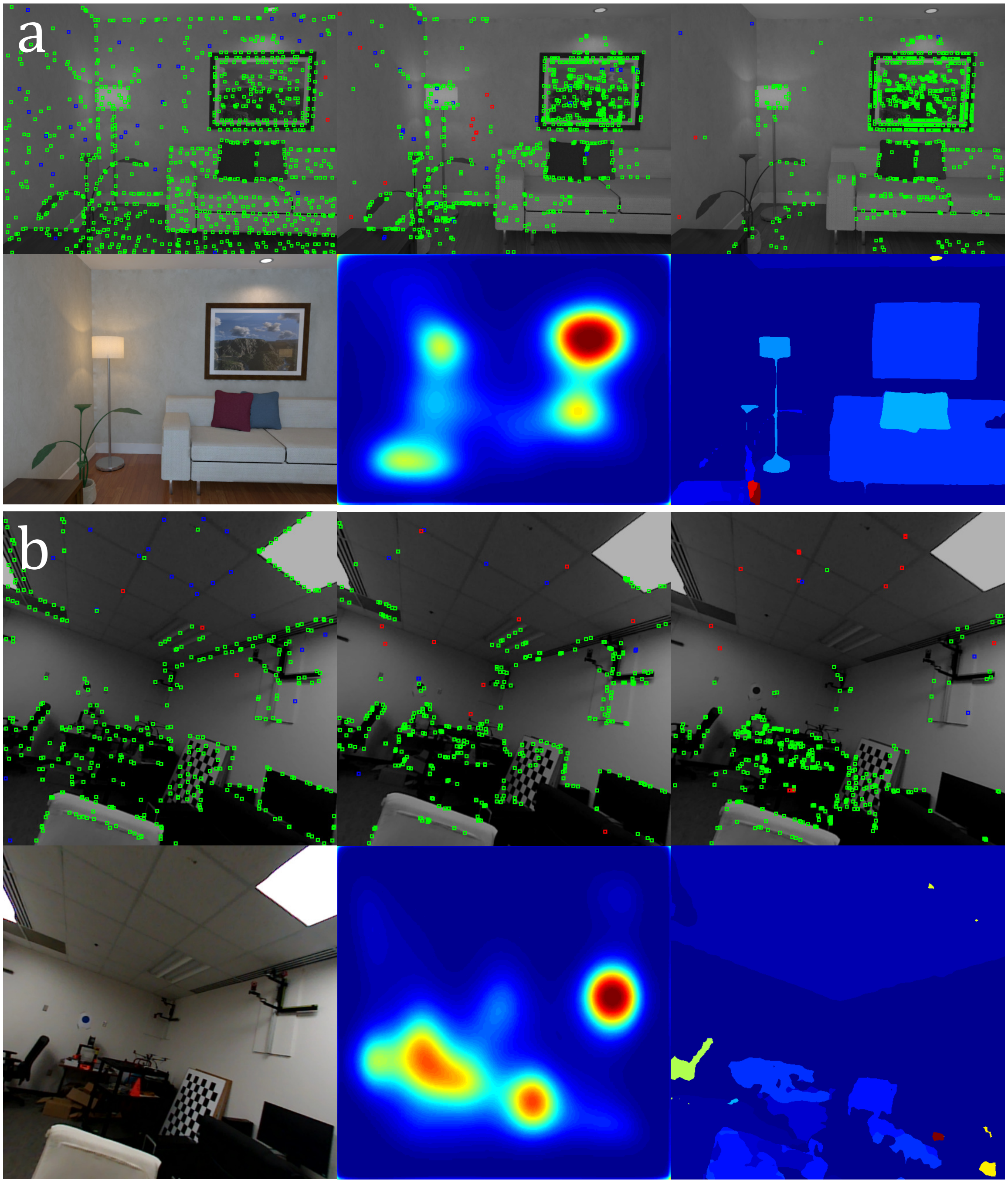

Figure 6.1: Point selection using different schemes. Top rows in (a) and (b), left to right: features selected using DSO's scheme, saliency only, saliency+scene parsing. Bottom rows in (a) and (b), left to right: input image, saliency, scene parsing output. Notice how using saliency+scene parsing removed all non-informative features.

Algorithm 2: Saliency based points selection.

```
Data: Desired number of points $N_{\text{des}}, s_{\text{smooth}}, \hat{S}^{\text{final}}$
Result: Selected points
Initialize selected point set as $\{\emptyset\}$, $N_{\text{sel}} = 0$;
while $N_{sel} < N_{des}$ do
    Randomly select a patch $M$ from distribution $P_S$;
    Split $M$ into $d \times d$ blocks;
    for each $4d \times 4d$ block do
        for each $2d \times 2d$ block do
            for each $d \times d$ block do
                Select a point with the highest gradient which surpass the
                  gradient threshold;
            end
            if no selected point in this block then
                Select a point with the highest gradient which surpass the
                  weaker gradient threshold;
            end
        end
        if no selected point in this block then
            Select a point with the highest gradient which surpass the much
              weaker gradient threshold;
        end
    end
    $N_{\text{sel}} = N_{\text{sel}} +$ the number of selected points;
end
```

Chapter 7: Results

In this chapter, we comprehensively evaluate SalientDSO on various datasets.

- **ICL-NUIM dataset** [58]: This dataset provides two scenes and four different trajectories for each scene which are obtained by running Kintinuous on real image data and finally used in a synthetic framework for obtaining ground-truth.

- **TUM monoVO dataset** [26]: This dataset provides 50 sequences comprising over 100 minutes videos. It ranges from indoor corridors to wide outdoor scenes. In our experiments, we only evaluate all methods on indoor sequences $\{\text{sequence\_}(1-18, 26, 28, 35-38, 40)\}$. Only the indoor sequences are chosen because the usage of saliency obtained by human gaze is meaningful only for indoor cluttered scenes.

- **CVL dataset**: This dataset was collected by the authors of this thesis is available at `prg.cs.umd.edu/SalientDSO.html`. The data was collected using a Parrot® SLAMDunk [59] sensor suite. The data from the left camera is used in the experiments.

Different parameters used for running the experiments are shown in Table. 7.1.

Table 7.1: Parameter settings for different datasets.

	TUM	ICL-NUIM	CVL
Num of active keyframes N_f	7	7	7
Num of active points N_p	2000	2000	1200
Global gradient constant g_{th}	7	3	7
Patch size K	8	8	8
Photometric correction	Yes	Not required	Not available

For ICL-NUIM dataset, photometric correction is not required. To comprehensively evaluate the proposed method, we run each sequence in both forward and backward direction 10 times.

7.1 Quantitative Evaluation

Fig. 7.1 shows the absolute trajectory Root Mean Square Error (RMSE_{ate}) on ICL-NUIM dataset. Using visual saliency driven features, SalientDSO performs better in accuracy as compared to DSO. We also report alignment error e_{align} on TUM monoVO dataset in Fig. 7.2. We disable the semantic filtering when we evaluate the proposed method on the TUM monoVO dataset, since this dataset provides only grayscale images and outputs from PSPNet are inaccurate and noisy for grayscale images. In Tables 7.2 and 7.3, we compare our method to DSO and ORB-SLAM on the ICL-NUIM and TUM monoVO datasets. DSO and ORB-SLAM are the current state-of-the-art direct and feature-based monocular VO methods. The results for DSO and ORB-SLAM are taken from [9]. ORB-SLAM is a full-fledged SLAM framework with loop closure and global alignment, while DSO and SalientDSO are merely odometry frameworks. To make the comparison fair, loop-closure detection and re-localization have been turned off for ORM-SLAM. The

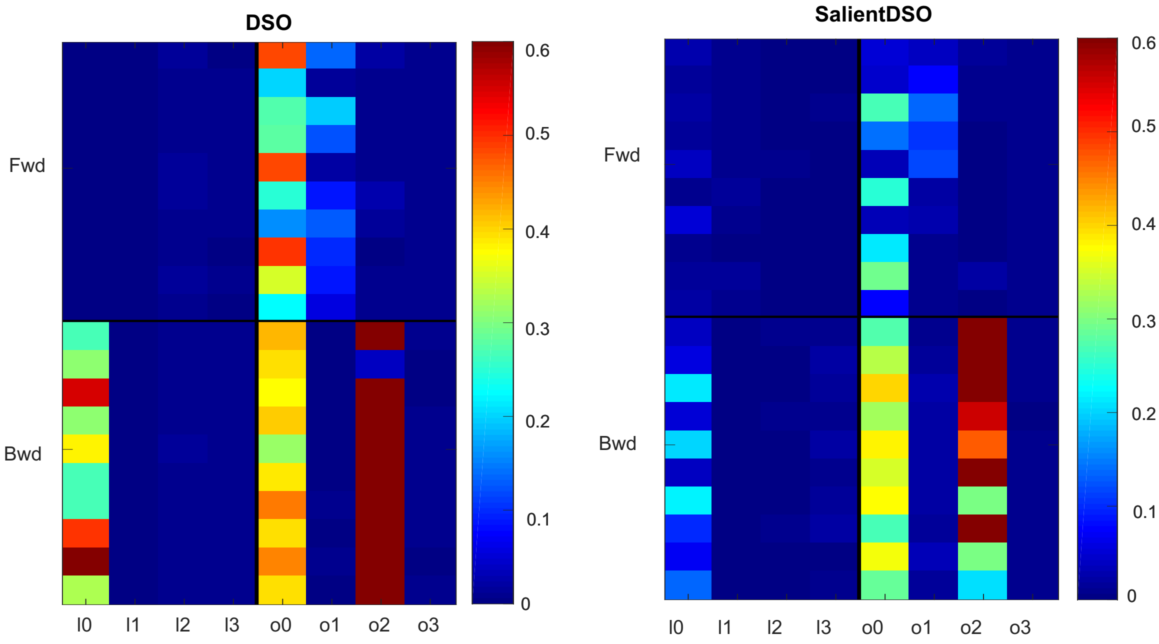

Figure 7.1: Comparison of evaluation results for ICL-NIUM dataset. Left: DSO, Right: SalientDSO. Each square correspondes to a color coded error. Note that Salient DSO almost always has lower error than it's DSO counterpart.

missing values in the table represent tracking failures. We achieve similar or better performance on most sequences. The improvement is not significant on the TUM monoVO dataset because most of the sequences involve a traversal through a hallway where there are no local salient objects or features for saliency prediction to work well. This makes SalientDSO's performance close to that of traditional DSO.

The claim in the thesis is that the usage of visual saliency should result in more robust features than just using image gradient based features as in DSO. The intuition behind this claim is that visual saliency includes high level semantics which inherently make the features more robust. To support this claim, we anticipate that SalientDSO should perform much better than DSO when the number of points is

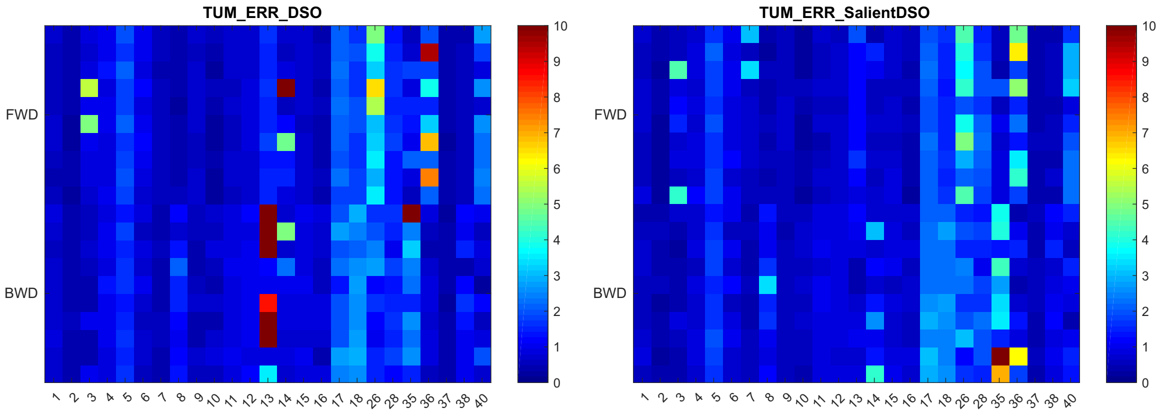

Figure 7.2: Comparison of evaluation results for TUM dataset. Left: DSO, Right: SalientDSO. Note that Salient DSO almost always has lower error than it's DSO counterpart. Note that, for the TUM dataset scene parsing was turned off as TUM dataset only provoides grayscale images and scene parsing outputs are very noisy for grayscale images.

very low (as low as 40 points). To demonstarate this claim, we evaluate on each CVL sequence. We run each sequence in both forward and backward direction 100 times, with an extremely low point density of $N_p = 40$. The results are shown in Table. 7.4. We define failure as either an optimization failure or tracking loss. Our proposed method is much more robust and predicts an accurate trajectory, while DSO has a much higher failure rate and its trajectory and projected point cloud shows significant drift in scale and position. An example of trajectory and projected point cloud is shown in Fig. 7.3. This experiment highlights the robustness of features chosen in SalientDSO for cluttered indoor scenes and how this will be useful for robots with very low computation power due to the less computational and memory requirements when N_p is low.

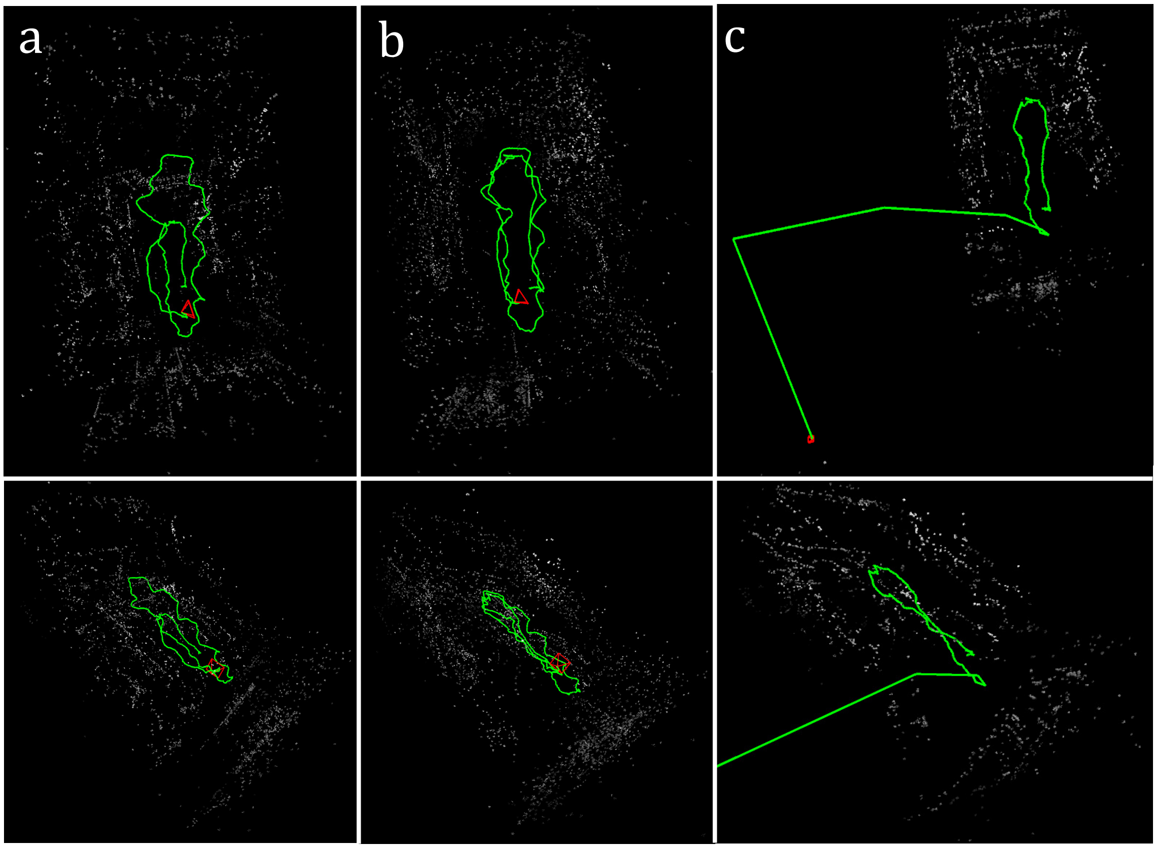

Figure 7.3: Comparison of outputs for $N_p = 40$ – very few features. (a) Success case of DSO with a large amount of drift, (b) Success case for SalientDSO, (c) Failure case of DSO where the optimization diverges due to very few features. Notice that SalientDSO can perform very well in these extreme conditions showing the robustness of the features chosen.

Table 7.2: RMSE_{ate} on ICL-NIUM dataset in m.

	Forward			Backward		
Sequence	ORB	DSO	SalientDSO	ORB	DSO	SalientDSO
ICL_l0	0.01	**0.003**	0.022	**0.01**	-	0.112
ICL_l1	0.02	**0.004**	0.009	0.04	0.003	**0.003**
ICL_l2	0.06	0.012	**0.004**	0.19	0.010	**0.005**
ICL_l3	0.03	0.006	**0.004**	0.05	**0.008**	0.013
ICL_o0	0.21	0.320	**0.140**	0.41	0.399	**0.336**
ICL_o1	0.83	0.094	**0.055**	0.68	**0.006**	0.020
ICL_o2	0.37	0.012	**0.008**	0.32	0.582	**0.512**
ICL_o3	0.65	**0.007**	0.009	0.06	**0.006**	0.008
Overall Avg.	0.271	0.057	**0.031**	0.218	0.144*	**0.126**

* indicates average taken only on sequences which completed.

7.2 Qualitative Evaluation

Examples of the reconstructed scenes of sequences CVL_01 and TUM seqence_01 are shown in Figs. 7.4 and 7.5 respectively. Although both reconstructed scenes look similar, one could observe that amount of drift in SalientDSO is much less compared to DSO (refer to the zoomed part of Fig. 7.4). One can clearly observe that the checkerboard of different loops align better in our approach. Instead of sampling random high gradient points, sampling salient and important points improves the robustness of VO. Sampling salient points achieves removing outliers and points with unconstrained depth in optimization which improves the prediction of initial estimates and the output of windowed bundle adjustment in optimization.

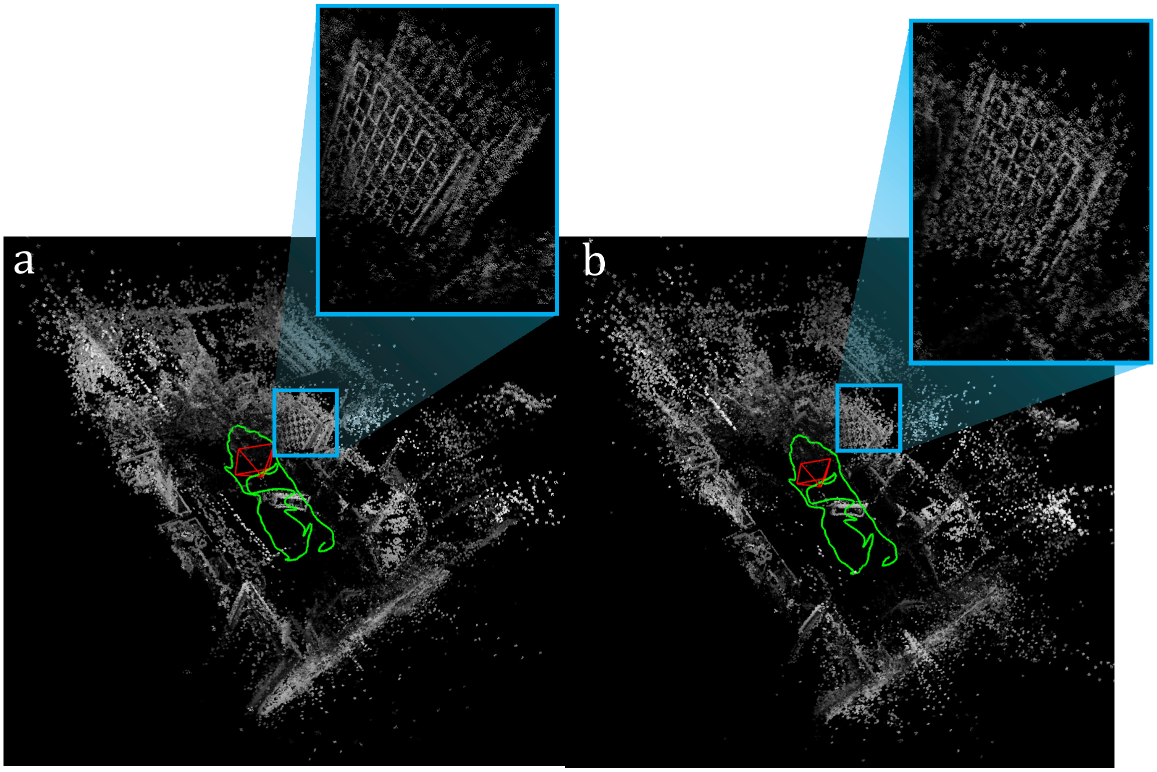

Figure 7.4: Comparison of drift. (a) DSO, (b) SalientDSO. Observe that SalientDSO's output has the checkerboard from different times more closely aligned as compared to DSO. Here $N_p = 1000$.

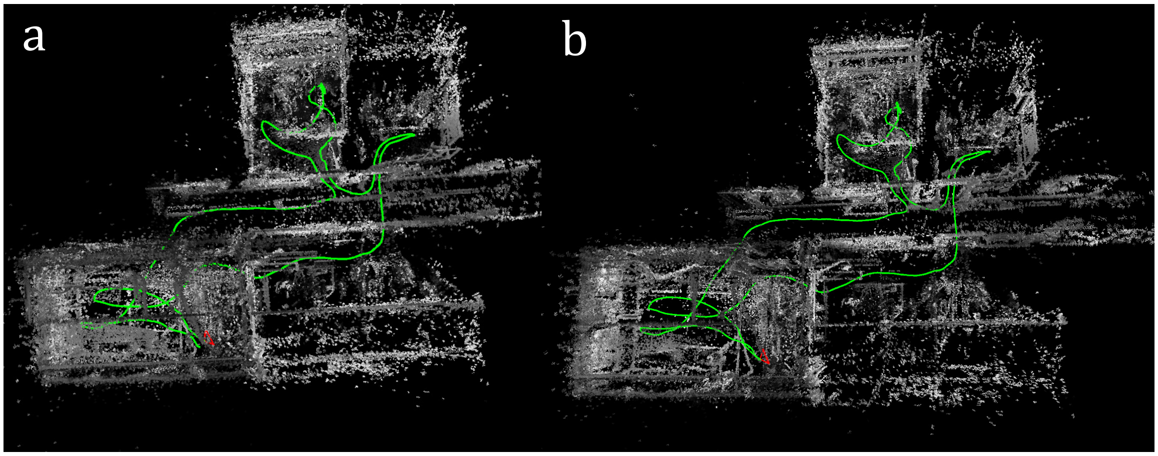

Figure 7.5: Sample outputs for TUM sequence_1. (a) DSO, (b) SalientDSO. Here $N_p = 1000$.

Table 7.3: e_{align} on TUM monoVO dataset in m.

	Forward			Backward		
Sequence	ORB	DSO	SalientDSO	ORB	DSO	SalientDSO
seq_01	3.02	**0.59**	0.60	1.73	0.72	**0.60**
seq_02	16.12	0.36	**0.33**	3.23	**0.43**	0.44
seq_03	3.42	1.75	**1.55**	1.42	0.59	**0.50**
seq_04	9.95	0.98	**0.82**	5.95	1.00	**0.76**
seq_05	-	1.86	**1.77**	-	**1.55**	1.66
seq_06	-	0.97	**0.93**	1.25	**0.73**	0.81
seq_07	1.69	**0.55**	1.14	2.02	**0.44**	0.48
seq_08	436.00	**0.36**	0.44	2.63	**1.28**	1.47
seq_09	2.04	0.65	**0.58**	0.67	**0.52**	0.53
seq_10	2.52	0.35	**0.34**	1.43	0.61	**0.61**
seq_11	7.20	0.62	**0.58**	2.99	**0.87**	0.89
seq_12	2.98	0.75	**0.67**	3.10	1.01	**0.84**
seq_13	5.13	1.54	**1.27**	2.59	8.96	**0.81**
seq_14	13.27	2.89	**0.71**	2.10	**1.35**	1.69
seq_15	2.90	0.71	**0.71**	1.90	0.88	**0.81**
seq_16	2.40	0.47	**0.45**	1.58	0.72	**0.67**
seq_17	12.29	2.10	**2.10**	**1.50**	2.13	2.50
seq_18	14.64	1.77	**1.52**	-	2.62	**2.47**
seq_26	28.46	3.98	**3.60**	4.62	**1.66**	1.89
seq_28	19.17	**1.48**	1.88	3.57	**1.47**	1.65
seq_35	14.09	1.10	**0.84**	16.81	**5.48**	9.97
seq_36	1.81	4.01	**3.25**	1.69	**0.70**	1.46
seq_37	0.60	**0.35**	0.40	1.30	**0.37**	0.46
seq_38	-	0.55	**0.50**	24.77	1.10	**1.03**
seq_40	-	**2.04**	2.16	18.93	**0.87**	1.04
Overall Avg.	28.55*	1.31	**1.17**	-	1.52	**1.44**

* indicates average taken only on sequences which completed.

Table 7.4: Comparison of success rate between DSO and SalientDSO on CVL dataset.

Sequence	DSO	SalientDSO
CVL_01_Fwd	53%	**65%**
CVL_01_Bwd	59%	**92%**
CVL_02_Fwd	73%	**96%**
CVL_02_Bwd	71%	**91%**

Chapter 8: Conclusion

We introduce the philosophy of attention and fixation to visual odometry. Based on this philosophy, we develop Salient Direct Sparse Odometry, which brings the concept of attention and fixation based on visual saliency into Visual Odometry to achieve robust feature selection. We provide thorough quantitative and qualitative evaluations on ICL-NUIM and TUM monoVO dataset to demonstrate that using salient features improves the robustness and accuracy. We also collect and publicly release a new CVL dataset with cluttered scenes for mapping. We show the robustness of our features by very low drift visual odometry with as low as 40 features per frame. Our method takes about a second per frame for computation of saliency and scene parsing on an NVIDIA Titan-Xp GPU and the remaining computations run real-time at 30fps on an Intel® Core i7 6850K 3.6GHz CPU. In the near future, we plan to extend our method to outdoor environment. We also consider to implement our method on hardware to make the complete pipeline real-time.

Bibliography

[1] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. Monoslam: Real-time single camera slam. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1052–1067, June 2007.

[2] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 225–234, Nov 2007.

[3] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. Dtam: Dense tracking and mapping in real-time. In *2011 International Conference on Computer Vision*, pages 2320–2327, Nov 2011.

[4] J. Engel, T. Schöps, and D. Cremers. LSD-SLAM: Large-scale direct monocular SLAM. In *European Conference on Computer Vision (ECCV)*, September 2014.

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

[6] Jan Stühmer, Stefan Gumhold, and Daniel Cremers. Real-time dense geometry from a handheld camera. In *Joint Pattern Recognition Symposium*, pages 11–20. Springer, 2010.

[7] Michael Bloesch, Sammy Omari, Marco Hutter, and Roland Siegwart. Robust visual inertial odometry using a direct ekf-based approach. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 298–304. IEEE, 2015.

[8] Sebastian A Scherer and Andreas Zell. Efficient onbard rgbd-slam for autonomous mavs. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 1062–1068. IEEE, 2013.

[9] J. Engel, V. Koltun, and D. Cremers. Direct sparse odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Apr 2017.

[10] Ajay Mishra, Yiannis Aloimonos, and Cheong Loong Fah. Active segmentation with fixation. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 468–475. IEEE, 2009.

[11] John Aloimonos et al. Active vision. *International journal of computer vision*, 1(4):333–356, 1988.

[12] Jeannette Bohg et al. Interactive perception: Leveraging action in perception and perception in action. *IEEE Transactions on Robotics*, 33(6):1273–1291, 2017.

[13] Ruzena Bajcsy et al. Revisiting active perception. *Autonomous Robots*, pages 1–20, 2017.

[14] Javier Civera, Dorian Gálvez-López, Luis Riazuelo, Juan D Tardós, and JMM Montiel. Towards semantic slam using a monocular camera. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 1277–1284. IEEE, 2011.

[15] Sean L Bowman, Nikolay Atanasov, Kostas Daniilidis, and George J Pappas. Probabilistic data association for semantic slam. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 1722–1729. IEEE, 2017.

[16] Lifeng An, Xinyu Zhang, Hongbo Gao, and Yuchao Liu. Semantic segmentation–aided visual odometry for urban autonomous driving. *International Journal of Advanced Robotic Systems*, 14(5):1729881417735667, 2017.

[17] Kostas Alexis Tung Dang, Christos Papachristos. Visual saliencyaware receding horizon autonomous exploration with application to aerial robotics. In *Robotics and Automation (ICRA), 2018 IEEE International Conference on*. IEEE, 2018.

[18] H. Jin, P. Favaro, and S. Soatto. Real-time 3d motion and structure of point features: a front-end system for vision-based control and interaction. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No.PR00662)*, volume 2, pages 778–779 vol.2, June 2000.

[19] Hailin Jin, Paolo Favaro, and Stefano Soatto. A semi-direct approach to structure from motion. *The Visual Computer*, 19(6):377–394, Oct 2003.

[20] R. Ranftl, V. Vineet, Q. Chen, and V. Koltun. Dense monocular depth estimation in complex dynamic scenes. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4058–4066, June 2016.

[21] Jan Stühmer, Stefan Gumhold, and Daniel Cremers. Real-time dense geometry from a handheld camera. In Michael Goesele, Stefan Roth, Arjan Kuijper, Bernt Schiele, and Konrad Schindler, editors, *Pattern Recognition*, pages 11–20, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[22] Levi Valgaerts, Andrés Bruhn, Markus Mainberger, and Joachim Weickert. Dense versus sparse approaches for estimating the fundamental matrix. *International Journal of Computer Vision*, 96(2):212–234, Jan 2012.

[23] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, Nov 2004.

[24] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In Aleš Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision – ECCV 2006*, pages 404–417, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[25] M. Pizzoli, C. Forster, and D. Scaramuzza. Remode: Probabilistic, monocular dense reconstruction in real time. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2609–2616, May 2014.

[26] J. Engel, V. Usenko, and D. Cremers. A Photometrically Calibrated Benchmark For Monocular Visual Odometry. *ArXiv e-prints*, July 2016.

[27] Stefan Leutenegger, Simon Lynen, Michael Bosse, Roland Siegwart, and Paul Furgale. Keyframe-based visualinertial odometry using nonlinear optimization. *The International Journal of Robotics Research*, 34(3):314–334, 2015.

[28] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1254–1259, Nov 1998.

[29] Jonathan Harel, Christof Koch, and Pietro Perona. Graph-based visual saliency. In *Proceedings of the 19th International Conference on Neural Information Processing Systems*, NIPS'06, pages 545–552, Cambridge, MA, USA, 2006. MIT Press.

[30] T. Judd, K. Ehinger, F. Durand, and A. Torralba. Learning to predict where humans look. In *2009 IEEE 12th International Conference on Computer Vision*, pages 2106–2113, Sept 2009.

[31] A. Borji. Boosting bottom-up and top-down visual features for saliency estimation. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 438–445, June 2012.

[32] Marcella Cornia, Lorenzo Baraldi, Giuseppe Serra, and Rita Cucchiara. A Deep Multi-Level Network for Saliency Prediction. In *International Conference on Pattern Recognition (ICPR)*, 2016.

[33] X. Huang, C. Shen, X. Boix, and Q. Zhao. Salicon: Reducing the semantic gap in saliency prediction by adapting deep neural networks. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 262–270, Dec 2015.

[34] S. Jetley, N. Murray, and E. Vig. End-to-end saliency mapping via probability distribution prediction. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5753–5761, June 2016.

[35] S. S. S. Kruthiventi, V. Gudisa, J. H. Dholakiya, and R. V. Babu. Saliency unified: A deep architecture for simultaneous eye fixation prediction and salient object segmentation. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5781–5790, June 2016.

[36] M. Kmmerer, L. Theis, and M. Bethge. Deep gaze i: Boosting saliency prediction with feature maps trained on imagenet. In *ICLR Workshop*, May 2015.

[37] M. Kümmerer, T. S. A. Wallis, and M. Bethge. DeepGaze II: Reading fixations from deep features trained on object recognition. *ArXiv e-prints*, October 2016.

[38] G. Li and Y. Yu. Visual Saliency Based on Multiscale Deep Features. *ArXiv e-prints*, March 2015.

[39] N. Liu and J. Han. A Deep Spatial Contextual Long-term Recurrent Convolutional Network for Saliency Detection. *ArXiv e-prints*, October 2016.

[40] Junting Pan, Elisa Sayrol, Xavier Giro-i Nieto, Kevin McGuinness, and Noel E. O'Connor. Shallow and deep convolutional networks for saliency prediction. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[41] E. Vig, M. Dorr, and D. Cox. Large-scale optimization of hierarchical features for saliency prediction in natural images. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2798–2805, June 2014.

[42] J. Pan, C. Canton Ferrer, K. McGuinness, N. E. O'Connor, J. Torres, E. Sayrol, and X. Giro-i-Nieto. SalGAN: Visual Saliency Prediction with Generative Adversarial Networks. *ArXiv e-prints*, January 2017.

[43] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.

[44] Ming Jiang, Shengsheng Huang, Juanyong Duan, and Qi Zhao. Salicon: Saliency in context. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[45] Zoya Bylinskii, Tilke Judd, Ali Borji, Laurent Itti, Frédo Durand, Aude Oliva, and Antonio Torralba. Mit saliency benchmark.

[46] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *ArXiv e-prints*, September 2014.

[47] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[48] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 740–755, Cham, 2014. Springer International Publishing.

[49] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 00, pages 6230–6239, July 2017.

[50] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[51] Bolei Zhou, Aditya Khosla, gata Lapedriza, Aude Oliva, and Antonio Torralba. Object detectors emerge in deep scene cnns. 12 2014.

[52] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June 2016.

[53] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs. *ArXiv e-prints*, December 2014.

[54] F. Yu and V. Koltun. Multi-Scale Context Aggregation by Dilated Convolutions. *ArXiv e-prints*, November 2015.

[55] Bryan C. Russell, Antonio Torralba, Kevin P. Murphy, and William T. Freeman. Labelme: A database and web-based tool for image annotation. *International Journal of Computer Vision*, 77(1):157–173, May 2008.

[56] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3485–3492, June 2010.

[57] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 487–495. Curran Associates, Inc., 2014.

[58] A. Handa, T. Whelan, J.B. McDonald, and A.J. Davison. A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM. In *IEEE Intl. Conf. on Robotics and Automation, ICRA*, Hong Kong, China, May 2014.

[59] Parrot SLAMDunk. http://developer.parrot.com/docs/slamdunk/, 2018.