Imperial College London

Final Year Project

Interim Report

Automated Photo-realistic Image Completion Using Dense Correspondence with Multiple Images

Author:
Youssef Rizk

Supervisor:
Prof. Pier-Luigi Dragotti

January 29, 2018

Imperial College London

Contents

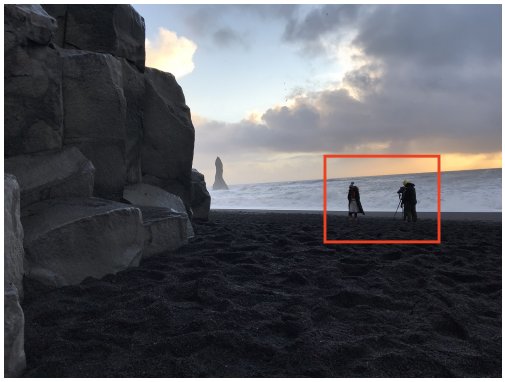

Figure 1: Photograph of Black Sand Beach occluded by tourists

1 Introduction

Image completion (image inpainting [1]) has been an old topic of interest for photographers and artists alike. The question of how to artificially, yet seamlessly, remove occlusions or restore missing areas to produce a realistically plausible photograph has been manually tackled with the use of tools like Photoshop, which have produced rather believable results. Of course, such techniques rely on exceptional skill and mastery of the software. With the modern proliferation of camera-enabled devices, an interest has spiked in the automation of such image completion tasks.

Certainly, in our daily lives, we often find that the pictures we take are, for one reason or another, occluded by some entity, be it a person, road works, or some other obstruction. To give a personal example, I recently visited Iceland, where I went to the Black Sand Beach in Vìk. I saw a great opportunity for an artistic photograph, and took the photo in Figure 1. Unfortunately, as it is a very touristic area, I was unable to capture a picture free of tourists (marked in the red box). Very inexperienced with photo-editing software, I am unable to manually enhance this picture. Automated image completion provides a remarkable solution to this, in that it abstracts from the user the workings of image editing to produce occlusion-free results.

Although there have been innumerable works tackling the problem of image completion, which will be reviewed in Section 4, the project outlined in this report specifically builds on the work presented in [2], in which the authors produce realistic, occlusion-free images by forming a dense corre-

spondence between the input and exemplar images. This project aims to build upon this framework by i) automatically choosing two (or more) exemplar images to be used in the image completion problem from a larger set of images and ii) extending the framework to use two (or more) exemplar images to produce realistically plausible photographs.

2 Problem Definition

The image completion problem can be formulated in terms of the known and occluded regions in an image. Let $\mathcal{O}$ and $\mathcal{S}$ denote the set of pixels, p, in the occluded and known parts of the image in question, respectively, where in principle $\mathcal{S}$ can originate from another image as will be seen shortly. In colored images, each pixel $p : (x, y) \in \mathbb{R}^2$ carries three color components defined in the RGB color space, (R, G, B) [3]. We define $\mathcal{F} : \mathbb{R}^2 \rightarrow \mathbb{R}^3$, which formalizes the relationship between a pixel and its corresponding color values. Assuming that $\mathcal{S}$ is taken only from the source image, we can define the occluded image in question, $\mathcal{I}$, such that

$$\mathcal{I} = \mathcal{F}(\mathcal{S}) \cup \mathcal{F}(\mathcal{O})$$

The problem of image completion attempts to estimate $\mathcal{F}(\mathcal{O})$ using information in $\mathcal{S}$ to construct an occlusion-free rendition of the original image. The quality of an attempt corresponds to how natural the result looks to a human viewer.

3 Project Specifications

In this section, I elaborate further on the expectations of the project, as well as user requirements. As previously mentioned, the main contributions of this project are to

- automatically select from a larger corpus of candidate images two exemplar images to be used for the image completion task
- extend the current framework to use the dense correspondences between the input image and the two exemplar images to construct an occlusion-free result

The only user inputs required are i) the occluded input image, and ii) the region of interest where the occlusion is present, which is indicated as a rectangular area, as in Figure 1. The algorithm will then find the two most suitable images from a stored collection to be used as exemplar images, and subsequently perform image completion. It may be more practical in future applications to also require the user to enter a keyword which describes the input image, such that we may query online search engines and retrieve more candidate exemplar images than those stored. Upon completion of the task, an occlusion-free image will be returned to the user.

4 Background

The background research section presented here aims to review past attempts at image completion, established methods for finding similar images, and the underlying framework in [2] upon which this project builds.

4.1 Image Completion

Two general families of image completion algorithms are diffusion-based and exemplar-based algorithms. Diffusion-based techniques propagate (or diffuse) image content inwards to the "hole" with smoothness constraints to achieve image completion [3]. The direction of propagation has been determined using various models. A typical approach was proposed by Bertalmio et al. [1], which iteratively propagates known image content into the "hole" along the direction of the lines of constant intensity (isophotes). Building upon this work, Bertalmio et al. [4] combine image inpainting techniques presented in [1] and texture synthesis to plausibly estimate the texture and structure present in the occluded region. Although decent results have been achieved using such techniques, the underlying drawback of diffusion-based algorithms is that they cannot plausibly complete larger occluded regions, as they only propagate local information to fill the hole.

The second category of image completion techniques is exemplar-based, which overcome this limitation, as they exploit the redundancies within the entire input image and use patches from the known regions of the image to fill in the "hole". The added benefit here over diffusion-based techniques is that exemplar-based methods do not restrict themselves to local information

around the "hole". Early advances in this area have typically used single-image completion, where information from the same image is used to fill in the hole. A seminal work in this area is [5], where Efros and Leung propose a texture synthesis technique that estimates the conditional probability of a pixel given its spatial neighbours by checking a sample image for similar neighborhoods. In essence, a missing pixel is copied from another pixel in the exemplar that has a similar neighborhood, where similarity is measured by a certain distance metric. To solve the image completion problem, Criminisi et al. [6] employ exemplar-based texture synthesis, combined with a best-first algorithm that completes structures followed by textures within the occluded region . In [4], the authors propose a novel framework that makes use of both diffusion & exemplar-based techniques. Their method decomposes an image into two functions which capture information about the underlying image structure and texture, respectively. Image inpainting and texture synthesis techniques are then used to complete these functions separately and the completed image is obtained by adding back these enhanced functions.

Indeed, similarly to [2], several works make use of patch-based approaches. In [7], Fang and Lien adopt a multiresolution, patch-based approach to solve the problem. The authors in [8] combine principles of patch-based synthesis with gradient domain blending and texture interpolation to create a framework that plausibly completes occluded images. Adopting an approach inspired by sparse representation, Xu and Sun [9] complete images by establishing a priority for the next patch to be filled in in the occluded region, based on the sparsity of its similarity with its spatial neighbours. The patch is then completed by using a sparse linear combination of candidate exemplar patches. Observing that the statistics of patch offsets are sparsely distributed, He and Sun [10] and Köppel et al. [11] use these statistics to greatly accelerate the matching process. It is worth mentioning that other techniques for solving the image completion task formulate the problem as a global optimization problem where a particular energy function is to be minimized. Works that follow this approach generally employ graph cuts [12] or belief propagation [13] to solve the optimization problem.

The aforementioned techniques, single-image exemplar and diffusion-based, suffer from the assumption that content required to fill in the occluded region in the source image can be found within the same image. This assumption does not always hold, for instance in cases where entire objects are occluded. Internet-based image completion algorithms seek to remedy

this shortcoming by completing the occluded region with content from similar images. The seminal work of Hays and Efros [14] set the scene for such algorithms. They propose an image completion technique whereby they find similar exemplar images from a large database of 2 million images, which are subsequently used to complete the source image. This is done through a form of graph cut seam finding and standard Poisson blending, and the user is presented with the 20 best completion results, where the ranking is expressed in terms of various matching distance metrics. Although the results are visually consistent, they are not often faithful to the real scene. In another attempt, Whyte et al. [15] use a Markov random field (MRF), which solves the labeling problem concerning which exemplar image should be used to fill in each pixel. In [16], the authors again obtain a ranked list of the most similar exemplars from a database. Starting with the highest ranked exemplar, they use its patches to complete the corresponding patches in the source image, employing the graph cut and Poisson blending technique, and then repeat the process for the non-completed patches in the source image using the next highest ranked image, and so on. In another attempt by Zhu et al. [17], given a small set of candidate exemplar images, point and line correspondences between each candidate image and the source image are established using a co-matching algorithm they developed. Each candidate image is then warped to the source image using a mesh-based warping algorithm, and the completion results are subsequently obtained through gradient-domain blending of the warped candidate into the occluded regions of the source. The best completion result is then ranked based on a score that accounts for warping and blending energies. Another notable contribution to the field is made by Wong and Orchard [18], where they attempt to overcome the risk that the exemplar image is itself occluded. The image completion task is tackled by finding the k-nearest neighbor patches in the known part of the image for a patch in the occluded region. The completion is then done by replacing the pixel in question with a weighted average of the nearest pixels, where the weights are reflective of the similarity between the patches.

4.2 Automatic Image Selection

This project aims to produce a tool to automatically determine two or more optimal candidate exemplar images from a larger set of images. Thus, in this subsection, I review some of the established techniques for determining similarity between images. In general, such approaches rely on first formulating a descriptor for the image and comparing it to other images through a particular distance metric.

In Internet-based image completion algorithms, the problem of finding the optimal exemplar images to be used in the completion step is frequently addressed. In the pioneering work of Hays and Efros [14], they employ the gist scene descriptor [19] to find semantically matching images to the source image, from a database containing 2 million images, subsequently using the 200 most similar scenes for the completion task. The 20 completion results that minimize a certain cost function the most are then presented to the user. In [20], Talat et al. represent the source and exemplar images using the gist scene descriptor, among other features, to determine image similarity. They propose a unified ranking algorithm in order to achieve value-invariance in the similarity measure to select the "top 1" exemplar candidate from a large corpus of images. Zhu et al. [17] make use of a two-stage filtering process to obtain exemplar images. The gist descriptor is initially used to filter out non-semantically similar images, followed by calculating a registration score that measures how similar the content in each image is to that in the source. Other notable approaches make use of the SIFT descriptor [21]. Amirshahi et al. [16] also apply a two-stage filtering process to rank optimal exemplar images. A first stage filter eliminates candidates with few SIFT keypoint matches, differing object scales, and varying image sizes. A second stage filter then ranks the candidate images according to the number of patches they can complete in the source image.

Other works approach this image similarity problem differently. In [22], the authors determine candidate images by checking the consistency between user-defined input image and available images, where a pairwise geometrical histogram is used to match the final candidate images. In [15], the exemplar images ("oracles") are chosen by geometrically then photometrically registering each image in the database with the source image. Geometric registration is performed using multiple homographies, and photometric registration is achieved using a global affine transformation on image intensities.

The given task is quite similar to that of object retrieval, where given a user-indicated region in an image the task is to find images portraying similar content. In [23], Sivic and Zisserman outline a Bag-of-Visual-Words approach by vector quantizing the descriptor vectors of the images in a database. An image is then represented using a Bag-of-Words (BoW), which can be used as an index for later querying. The authors in [24] enhance the standard BoW approach by noting that word similarity should be based on their association with the same object, as well as their proximity in the feature space. The concept of Blobworld [25] has been proposed by Belongie et al. to tackle this problem. The technique operates by grouping pixels according to their low-level coherency to make up an object or part of an object (a blob), followed by describing them in terms of their texture and anisotropy. Images which contain similar blobs can then be efficiently searched in a database. With focus particularly in the object retrieval domain, Sünderhauf and Protzel propose to determine image similarity by comparing the BRIEF-Gist descriptor [26], which essentially extends BRIEF for use as a holistic image descriptor. An image is partitioned into $n \times n$ tiles, which are then downsampled to a BRIEF patch size (e.g. 60) and BRIEF is calculated for each. The resulting descriptors can be calculated and compared very efficiently. Although [26] achieves remarkable performance in terms of accuracy and computational speed relative to other descriptors like SIFT and SURF [27], it is vulnerable to viewpoint changes. Realizing this, the authors in [28] remedy this by extracting three BRIEF-Gist descriptors for each image, one for the original image, one for the leftmost 80%, and one for the rightmost 80%.

4.3 Underlying Framework

In this subsection, I present the methods used in [2] to tackle the image completion problem. This work falls under the Internet-based image completion category. The image completion problem is approached by forming a dense correspondence between the source image and an exemplar image. This manifests advantages over methods that employ sparse correspondence in that it i) is more sensitive to deformations which may be missed by sparse methods and ii) provides a rich set of color correspondences which can later be used for color correction. This paper makes similar assumptions to past works, namely that the region of interest (ROI) is specified by the user, and that there are many images available on the Internet which are similar to

the source image, obtainable from standard search engines (e.g. Google). The dense correspondence is estimated between the source and exemplar images for each level in an image pyramid, following a coarse-to-fine order. An expectation-maximization (EM) approach is used to jointly estimate inliers/outliers and dense correspondence interpolation. The model parameters and correspondence are then used to initialize the next pyramid level. After the occluded region is filled with the corresponding pixels from the exemplar, it is natural to note a color discrepancy, and consequently a color correction algorithm is applied. The following aims to further clarify some of the concepts behind the algorithm.

Definitions and Notations

Let us denote with $\mathcal{I}^1$ and $\mathcal{E}^1$ the source and exemplar images, respectively. We create two image pyramids from $\mathcal{I}^1$ and $\mathcal{E}^1$, namely $\{\mathcal{I}^k\}_{k=1}^K$ and $\{\mathcal{E}^k\}_{k=1}^K$, where each level is scaled down by a factor of 2^{k-1}. The size of the image pyramid K is the largest integer such that $2^{K-1} \times \max(m,n) \geq 32$, where $\mathcal{I}^1 \in \mathbb{R}^{m \times n}$. Now, we denote with $\mathcal{I}_r^k(\boldsymbol{p})$ the $(2r+1) \times (2r+1)$ patch centered on $\boldsymbol{p}$ in $\mathcal{I}^k$.

We denote with $\mathcal{F}^k$ the nearest neighbour field (NNF) at the kth level, which relates every patch in $\mathcal{I}^k$ to its nearest neighbor (NN) patch in $\mathcal{E}^k$. As the two images may vary in viewpoint, there are $D = 4$ degrees of freedom in the matching parameters of the NNF, namely position (u, v), scale s, and orientation θ. $\mathcal{F}^k(\boldsymbol{p}) = (u, v, s, \theta)$ denotes the matching parameter of $\mathcal{I}_r^k(\boldsymbol{p})$ in $\mathcal{E}^k$. Finally, $\mathcal{E}^k(\mathcal{F}^k(\boldsymbol{p}))$ then denotes the NN patch on $\mathcal{E}^k$ at position (u, v), patch radius $s \times r$, and orientation θ.

PatchMatch

In order to form the NNF, this work uses the PatchMatch [29] [30] algorithm, which will be reviewed here, along with the modifications made to suit this application. PatchMatch attempts to quickly calculate the NNF through a randomized approach, and it consists of three steps: initialization, propagation, and random search.

In initialization, the NNF can be initialized either randomly or using some prior information. In the case of random initialization, the correspondences are chosen from an independent uniform distribution across the full

range of the exemplar image.

The next stage of PatchMatch uses an iterative update process to fine-tune the correspondences. Since images are highly structured, good matching parameters can be used to update those of their spacial neighbours $\Psi_{\mathbf{p}}$ with minor adjustments. This makes up the propagation step. As an example, the patch $\mathcal{I}_r^k(\boldsymbol{p})$ can update its own matching parameter in the NNF, $\mathcal{F}^k(\boldsymbol{p})$ by using those of its spacial neighbours $\Psi_{\mathbf{p}}$ as follows:

$$\mathcal{F}^k(\boldsymbol{p}) = \underset{\mathcal{F}^k(\boldsymbol{p_i})}{\arg\min}\{D(\mathcal{F}^k(\boldsymbol{p_i})) \mid \boldsymbol{p_i} \in \boldsymbol{p} \cup \Psi_{\mathbf{p}}\} \tag{1}$$

where we define $D(\mathcal{F}^k(\boldsymbol{p_i}))$ as a distance function between $\mathcal{I}_r^k(\boldsymbol{p})$ and $\mathcal{E}^k(\mathcal{F}^k(\boldsymbol{p})+\omega)$, with ω being the affine adjustment. Certainly, the way in which we define $\Psi_{\mathbf{p}}$ is crucial to how the propagation step will operate. Propagation is carried out in four directions [31], namely

$$\Psi_{\mathbf{p}} = \{\boldsymbol{p} - \mathbf{1}_h, \boldsymbol{p} - \mathbf{1}_v\} \qquad \Psi_{\mathbf{p}} = \{\boldsymbol{p} + \mathbf{1}_h, \boldsymbol{p} + \mathbf{1}_v\}$$
$$\Psi_{\mathbf{p}} = \{\boldsymbol{p} + \mathbf{1}_h, \boldsymbol{p} - \mathbf{1}_v\} \qquad \Psi_{\mathbf{p}} = \{\boldsymbol{p} - \mathbf{1}_h, \boldsymbol{p} + \mathbf{1}_v\}$$

After propagation is completed at every position, random search is applied to prevent matching parameters in the NNF from being stuck in local minima. Several matching parameters around $\mathcal{F}^k(\boldsymbol{p})$ in the parameter space are tested to see if they produce a lower distance to the corresponding patch in the source image. If we define $\boldsymbol{v_0} = \mathcal{F}^k(\boldsymbol{p})$, then we sample $\boldsymbol{u_i} = \boldsymbol{v_0} + w\alpha^i\mathbf{R_i}$, where w is a maximum search radius, α is a fixed ratio between successive search windows, and $\mathbf{R_i}$ is a uniform random variable $[-1, 1] \times [-1, 1]$. In this work, $\alpha = 1/2$ and the random search continues until the search radius is below 1 pixel. Such a random search is often inefficient as it uses the same maximum search radius for all patches, and consequently, this work presents an adaptive random search which adaptively selects candidate matching parameters based on their reliability. In essence, a reference NNF is derived from the interpolation function of the previous level to guide the random search. If a randomly generated parameter is too far from the reference NNF, it is unlikely to be a good candidate, and thus random search can be terminated for unreliable matching parameters.

In [29], the convergence of PatchMatch has been shown. However, it has also been noted that it serves greatly when only approximate patch

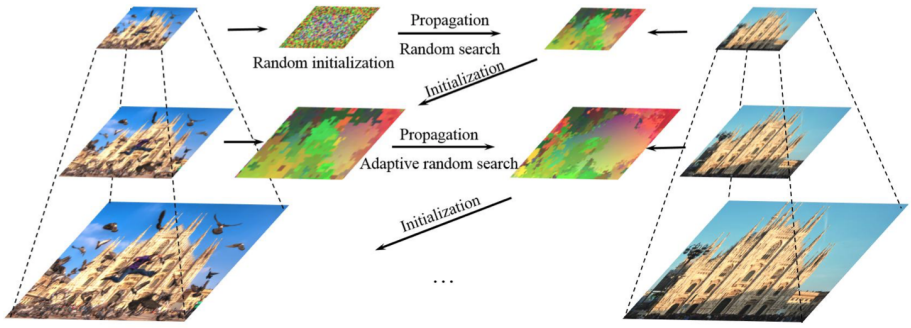

Figure 2: Operation of PatchMatch, where we establish correspondences for the coarsest level and use generated results as initializations for subsequent levels (obtained from [2])

matches are required, in which case the number of iterations of PatchMatch can be limited. In this work, a hierarchical PatchMatch is adopted, where we randomly initialize the coarsest level of the image pyramids and use the generated matching parameters as the initialization for the subsequent levels, following a coarse-to-fine scheme. This scheme is further demonstrated in Figure 2.

Feature Representation and Distance Metric

Each patch is represented using the BRIEF descriptor [32]. This approach is inspired by earlier work that showed that patches could be accurately classified using a small number of pair-wise intensity comparisons. Formally, we define a binary test τ on a patch as

$$\tau(c, \boldsymbol{p}, \boldsymbol{q}) = \begin{cases} 1 & c(\boldsymbol{p}) < c(\boldsymbol{q}), \\ 0 & \text{otherwise}, \end{cases}$$

where c is a randomly chosen feature channel, and $\boldsymbol{p}$ and $\boldsymbol{q}$ are randomly sampled points within the patch. There will be n_b tests for each descriptor (e.g. 128, 256, 512), and once they have been determined, all patches will use the same tests. Thus the descriptor of a patch can then be described as an n_b-bit binary string, which is the concatenation of the results of the tests, and is the binary counterpart of $\sum_{i=1}^{n_b} 2^{i-1}\tau(c_i, \boldsymbol{p}_i, \boldsymbol{q}_i)$. Such a formulation is

very suitable for use with the Hamming Distance as the distance metric D in Eqn.(1), and this strategy is adopted in the paper.

Now, to achieve color invariance, the RGB color images are transformed to the CIE La*b* color space, where L is the illuminance, and a* and b* are chrominance channels. To account for the edges, two gradient channels are also included, namely $\nabla_{\mathbf{x}}\mathrm{L}$ and $\nabla_{\mathbf{y}}\mathrm{L}$ and thus the final feature channel set is $\mathcal{C} = (\mathrm{L}, \mathrm{a}^*, \mathrm{b}^*, \nabla_{\mathbf{x}}\mathrm{L}, \nabla_{\mathbf{y}}\mathrm{L})$. In other words, in each of the binary tests, c_i is chosen from this feature channel set.

Nearest Neighbor Field Interpolation

The acquired NNF using the basic PatchMatch with the BRIEF descriptor is still relatively noisy and with a large region of outliers in the occluded part of the image. To plausibly transfer corresponding pixels from the exemplar to the source image, the NNF needs to be interpolated, which is achieved with an EM approach that jointly estimates inliers/outliers and interpolates the NNF over the occluded region.

To begin, we assume there are N matched pixel-to-pixel correspondences generated by the basic PatchMatch within the ROI of pyramid level k, $\{(\mathbf{p_i}, \mathcal{F}^k(\mathbf{p_i}))\}_{i=1}^N$. For simplicity of notation, we let $(\boldsymbol{x_i}, \boldsymbol{y_i}) = (\boldsymbol{p_i}, \mathcal{F}^k(\boldsymbol{p_i}))$. We also denote with $\mathbf{X} = (\boldsymbol{x_1}, ..., \boldsymbol{x_n})^T \in \mathbb{R}^{N\times 2}$ and $\mathbf{Y} = (\boldsymbol{y_1}, ..., \boldsymbol{y_n})^T \in \mathbb{R}^{N\times D}$. We further assume that the generated NNF consists of a mixture of Gaussian distributed inliers and uniformly distributed outliers, and that the components of the NNF are independent. We assume the fitting error of the inliers follows a Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$ with zero mean and variance $\mathbf{\Sigma} \in \mathbb{R}^{D\times D}$. We define the outliers as the data pairs which cannot be well defined by the NNF interpolation function $\boldsymbol{f}$. To describe inliers and outliers, we define an indicator $z_i \in \{0, 1\}$ for each data pair $(\boldsymbol{x_i}, \boldsymbol{y_i})$ to describe whether the data pair is an inlier ($z_i = 1$) or outlier ($z_i = 0$). Given the aforementioned assumptions, the likelihood function can be expressed as

$$p(\mathbf{Y} \mid \mathbf{X}, \boldsymbol{\theta}) = \prod_{i=1}^{N} \Big(\gamma \frac{\exp(-d_i)}{\sqrt{\det(2\pi\mathbf{\Sigma})}} + \frac{1-\gamma}{V}\Big), \tag{2}$$

where $\boldsymbol{\theta} = \{\boldsymbol{f}, \gamma, \mathbf{\Sigma}\}$ is the model parameter, γ is the percentage of inliers, $d_i = \frac{1}{2}(\boldsymbol{y_i} - \boldsymbol{f}(\boldsymbol{x_i}))^T\mathbf{\Sigma}^{-1}(\boldsymbol{y_i} - \boldsymbol{f}(\boldsymbol{x_i}))$, and V is the volume of the NNF parameter space. We now assume a smooth prior on the interpolation function as

$p(\boldsymbol{f}) \propto \exp(-\frac{\lambda}{2}\phi(\boldsymbol{f}))$, where $\phi(\boldsymbol{f})$ is a smoothness function and λ is a regularization parameter. Using this, the posterior distribution of the model parameter can be estimated with Bayes rule as $p(\boldsymbol{\theta} \mid \mathbf{X}, \mathbf{Y}) \propto p(\mathbf{Y} \mid \mathbf{X}, \boldsymbol{\theta})p(\boldsymbol{f})$. The optimal model parameter $\theta^\star$ is then obtained from a Maximum A Posteriori (MAP) of θ, i.e.

$$\theta^\star = \arg\max_{\theta} p(\mathbf{Y} \mid \mathbf{X}, \boldsymbol{\theta})p(\boldsymbol{f}), \tag{3}$$

We now consider the operation of the EM algorithm, which is applied to iteratively estimate the model parameter $\boldsymbol{\theta}$. We first define some new quantities, namely $p_i = p(z_i = 1 \mid \boldsymbol{x_i}, \boldsymbol{y_i}, \boldsymbol{\theta})$ and $P = \sum_{i=1}^{N} p_i$. Now, substituting Eqn. 2 into Eqn. 3, along with some manipulation, we can obtain the negative log-likelihood function

$$Q(\boldsymbol{\theta}) = \sum_{i=1}^{N} p_i d_i + \frac{DP}{2} \ln\det(\boldsymbol{\Sigma}) - P\ln\gamma - (N-P)\ln(1-\gamma) + \frac{\lambda}{2}\phi(\boldsymbol{f}), \tag{4}$$

In the $t+1$th iteration, the EM algorithm completes an expectation followed by a maximization step. During the expectation step, the algorithm calculates the posteriori probability of $p(z_i = 1 \mid \boldsymbol{x_i}, \boldsymbol{y_i}, \boldsymbol{\theta_t})$ using the previous iteration's model parameter $\boldsymbol{\theta_t} = \{\boldsymbol{f}, \gamma_t, \boldsymbol{\Sigma_t}\}$. In our case, p_i can be estimated through

$$p_i = \frac{\gamma_t \exp(-d_{i,t})}{\gamma_t \exp(-d_{i,t}) + \frac{1-\gamma}{V}\sqrt{\det(2\pi\boldsymbol{\Sigma_t})}}, \tag{5}$$

In the maximization step, the algorithm subsequently updates the model parameter such as to minimize the negative log-likelihood function. For γ_{t+1} and $\boldsymbol{\Sigma_{t+1}}$, they can be calculated as follows

$$\gamma_{t+1} = \frac{\mathrm{Tr}(\mathbf{P})}{N}, \tag{6}$$

$$\boldsymbol{\Sigma}_{t+1} = \frac{(\mathbf{Y}(:,i) - \mathbf{Z}(:,i))^T \times \mathbf{P} \times (\mathbf{Y}(:,i) - \mathbf{Z}(:,i))}{\mathrm{Tr}(\mathbf{P})}, \tag{7}$$

where $\mathbf{P} = \mathrm{diag}(p_1, ..., p_N)$ is an $N \times N$ diagonal matrix with diagonal entries of $(p_1, ..., p_N)$, $\mathbf{Z} = (\boldsymbol{f}_t(\boldsymbol{x_1}), ..., \boldsymbol{f}_t(\boldsymbol{x_N}))^T \in \mathbb{R}^{N \times D}$ are the estimated NNF using $\boldsymbol{f}_t$, and $\mathbf{Y}(:,i)$ is the ith column of $\mathbf{Y}$.

Finding an expression for $\boldsymbol{f}_{t+1}$ is slightly more involved, and we proceed as follows. Using the terms in Eqn. 4, we define an energy function $E(\boldsymbol{f})$ as

$$E(\boldsymbol{f}) = \frac{1}{2}\sum_{i=1}^{N} p_i(\boldsymbol{y_i} - \boldsymbol{f}(\boldsymbol{x_i}))^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{y_i} - \boldsymbol{f}(\boldsymbol{x_i})) + \frac{\lambda}{2}\phi(\boldsymbol{f}), \tag{8}$$

In order to find $\boldsymbol{f}_{t+1}$, we minimize $E(\boldsymbol{f})$ with respect to $\boldsymbol{f}$. To do this, we define the smoothness function $\phi(\boldsymbol{f}) = ||\boldsymbol{f}||_{\mathcal{H}}$, where $\mathcal{H}$ is a reproducing kernel Hilbert space (RKHS) and is chosen as Gaussian in this paper. Using the representer theorem [33], we can express $\boldsymbol{f}$ as a weighted sum of kernel products, namely

$$\boldsymbol{f}(\boldsymbol{x}) = \sum_{i=1}^{N} k(\boldsymbol{x}, \boldsymbol{x}_n)\boldsymbol{w}_n, \tag{9}$$

where $k(\boldsymbol{a}, \boldsymbol{b}) = \exp\big(-\frac{||\boldsymbol{a}-\boldsymbol{b}||^2}{\beta}\big)$ is a reproducing kernel with filter range β, and $\boldsymbol{w}_n \in \mathbb{R}^4$ is the weight associated with $k(., \boldsymbol{x}_n)$. We can vectorize the formulation in Eqn. 8 by defining two new quantities: $\widetilde{\mathbf{W}} = (\boldsymbol{w}_1^T, ..., \boldsymbol{w}_N^T)^T \in \mathbb{R}^{ND\times 1}$ which is the coefficient column vector, and $\mathbf{K} \in \mathbb{R}^{ND\times ND}$ which is an $N \times N$ block matrix with (i, j)th block being $D \times D$ with entries corresponding to $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$. Using these quantities, we can now reformulate Eqn. 8 and obtain the following closed-form solution for $\widetilde{\mathbf{W}}$,

$$\widetilde{\mathbf{W}} = (\mathbf{K} + \lambda\widetilde{\mathbf{P}}^{-1})^{-1}\widetilde{\mathbf{Y}}, \tag{10}$$

where $\widetilde{\mathbf{Y}} = (\boldsymbol{y}_1^T, ..., \boldsymbol{y}_N^T)^T \in \mathbb{R}^{ND\times 1}$ is a column vector, and $\widetilde{\mathbf{P}} = \mathbf{P} \otimes \Sigma^{-1}$ ($\otimes$ denoting the Kronecher product). As calculating the closed-form solution above is quite computationally expensive, a fast approximation is used [34], [35], whereby $\widetilde{\mathbf{W}}$ is determined using only a subset of M control points $\{\tilde{\boldsymbol{x}}_{\boldsymbol{i}}\}_{i=1}^{M}$ (which are chosen with high probability of being inliers) with $M << N$. This reduces the size of $\widetilde{\mathbf{W}}$ from $N \times N$ to $M \times M$, thereby also reducing the complexity from $O(N^3)$ to $O(M^3)$. In this case, we reformulate the interpolation function as $\boldsymbol{f}(\boldsymbol{x}) = \sum_{m=1}^{M} k(\boldsymbol{x}, \widetilde{\boldsymbol{x}}_{\boldsymbol{m}})\boldsymbol{w}_m$. The coefficient matrix $\widetilde{\mathbf{W}}$ is then redefined as

$$\widetilde{\mathbf{W}} = (\widetilde{\mathbf{U}}^T\widetilde{\mathbf{P}_\mathbf{I}}\widetilde{\mathbf{U}} + \lambda\widetilde{\mathbf{Q}})^{-1}\widetilde{\mathbf{U}}^T\widetilde{\mathbf{P}_\mathbf{I}}\widetilde{\mathbf{Y}}, \tag{11}$$

with $\widetilde{\mathbf{U}} \in \mathbb{R}^{ND\times MD}$ being a $N \times M$ block matrix with the (i, j)th block $\in \mathbb{R}^{D\times D}$ having entries corresponding to $k(\boldsymbol{x}_i, \widetilde{\boldsymbol{x}}_j)$, $\widetilde{\mathbf{P}_\mathbf{I}} = \mathbf{P} \otimes \mathbf{I}$, $\widetilde{\mathbf{Q}} = \mathbf{Q} \otimes \boldsymbol{\Sigma}$

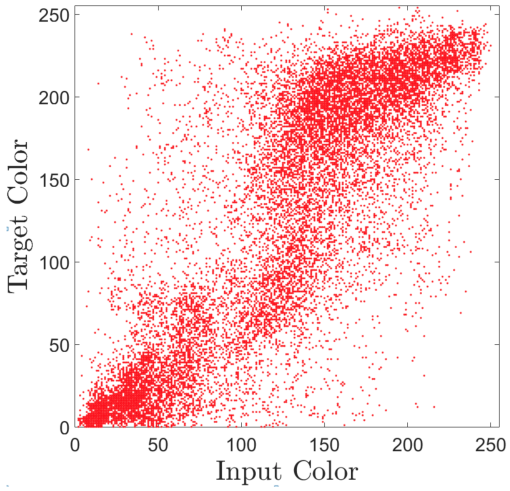

Figure 3: Typical Noisy Color Correspondences (obtained from [2])

($\mathbf{Q} \in \mathbb{R}^{M \times M}$ being the inter Gram matrix and $\mathbf{Q}(i,j) = k(\widetilde{\boldsymbol{x}}_i, \widetilde{\boldsymbol{x}}_j)$). Thus, the interpolation function $\widetilde{\boldsymbol{f}}$ is defined through the selected M control points and the coefficient matrix $\widetilde{\mathbf{W}}$. The EM algorithm iterates until the negative log-likelihood function converges or the maximum iteration $\mathcal{T}_{\text{EM}}$ is reached.

Image Completion

Upon completion of PatchMatch and NNF Interpolation, a smooth NNF $\mathcal{F}$ is obtained through

$$\mathcal{F} = \mathbf{W}\mathbf{U}^{\mathbf{T}}, \tag{12}$$

where $\mathbf{W} = (\boldsymbol{w}_1, ..., \boldsymbol{w}_M) \in \mathbb{R}^{D \times M}$ is the interpolation coefficient matrix, and $\mathbf{U} \in \mathbb{R}^{N \times M}$ is the inter Gram matrix where $\mathbf{U}(i,j) = k(\boldsymbol{x}_i, \tilde{\boldsymbol{x}}_j)$. To actually complete the image, we define a mask $\mathcal{M}$ which takes on a value of 1 on pixels with $p_i \geq \tau_p$ ($\tau_p = 0.7$), and 0 otherwise, and we erode this mask a few times. We define with $\mathcal{M}_i$ the set of pixels with values equal to i $\forall i \in \{0, 1\}$. For all pixels in $\mathcal{M}_0$, we replace the pixel $\boldsymbol{p}$ with its corresponding pixel value in $\mathcal{E}^1(\mathcal{F}(\boldsymbol{p}))$. We denote the completed image with $\mathcal{I}^c$.

Color Correction

Since there are likely to be variations in the conditions during which each picture was taken, it is natural to expect a color difference between the source and exemplar images. In order to overcome the color discrepancies of the interpolated result, a color correction algorithm is applied. It is worth noting the domains in which each of the steps in this framework are applied. We note that the PatchMatch and NNF Interpolation steps are conducted in the CIE La*b* color space in order to achieve color invariance. The completed image is then constructed in the RGB color space, where we subsequently also apply the color correction algorithm.

Unlike other color correction methods, we use the fact that the previously computed dense correspondence simultaneously provides us with pixel-wise color correspondences within $\mathcal{M}_1$, denoted by $\mathcal{D} = \{(x_{ci}, y_{ci})\}_{i=1}^{|\mathcal{M}_1|}$, where x_{ci} and y_{ci} are the corresponding color values on the input and exemplar image, respectively, and c denotes a color channel. For each RGB color channel, we fit a color transfer curve using the correspondences in $\mathcal{D}$ and apply them to $\mathcal{M}_0$ within $\mathcal{I}_c$. The color transfer curve $\boldsymbol{f}_c$ is modeled as a piece-wise cubic spline with L knots

$$\boldsymbol{f}_c(x) = \sum_{i=1}^{L} c(i)B(x-i), \tag{13}$$

where $c(i)$'s are the B-spline coefficients, and $B(x)$ is a cubic B-spline basis function.

Problematically, however, the color correspondence are often noisy, as can be seen in Figure 3. Although the obtained dense correspondence may be accurate, this noise could be a result of subtle differences between the two images, such as shadows, or inherently noisy images, where a single color corresponds to multiple colors. This must be addressed before we fit the B-spline curve, as this will greatly affect the quality of the color transfer. Again, we adopt an EM approach, similarly to the NNF interpolation. To do this, we reformulate the color correspondence as $\mathcal{D} = \mathcal{D}_0 \cup \mathcal{D}_1 \cup ... \cup \mathcal{D}_{255}$, where $\mathcal{D}_m = \{m, y_{m,i}\}_{i=1}^{N_m}$ for $m = 0, 1, 2, ..., 255$. We note that $\mathcal{D}_m$ can be modeled as a mixture of Gaussian distributed inliers and uniformly distributed outliers. The model parameters $\boldsymbol{\theta}_m$ that we are estimating here are percentage of inliers γ_m and variance of inliers σ_m^2.

In the expectation step, we calculate the probability that a color pair

$(m, y_{m,i})$ is an inlier through

$$p_{m,i} = \frac{\gamma_m \exp\big(-\frac{||y_{m,i}-\boldsymbol{f}_c(m)||^2}{2\sigma_m^2}\big)}{\gamma_m \exp\big(-\frac{||y_{m,i}-\boldsymbol{f}_c(m)||^2}{2\sigma_m^2}\big) + \frac{1-\gamma_m}{256}(2\pi\sigma_m^2)^{1/2}}. \tag{14}$$

In the maximization step, the model parameters are updated according to the following:

$$\gamma_m = \frac{\mathrm{Tr}(\mathbf{P}_m)}{N_m}, \tag{15}$$

$$\sigma_m^2 = \frac{(\mathbf{Y}_m - \mathbf{Z}_m)^T \times \mathbf{P}_m \times (\mathbf{Y}_m - \mathbf{Z}_m)}{\mathrm{Tr}(\mathbf{P}_m)}, \tag{16}$$

where $\mathbf{P}_m = \mathrm{diag}(p_{m,1}, ..., p_{m,N_m})$, $\mathbf{Z}_m = (\boldsymbol{f}_c(m), ..., \boldsymbol{f}_c(m))^T \in \mathbb{R}^{N_m}$, and $\mathbf{Y}_m = (y_{m,1}, ..., y_{m,N_m})^T$.

The EM algorithm is applied for $\mathcal{T}_{\text{in}}$ iterations for each color $m \in \{0, ..., 255\}$, followed by removing the color pairs with low probability of being an inlier, where this low probability threshold is set as 0.7. Once the outliers have been removed, we fit a B-spline curve with the updated color correspondences. After iterating this process $\mathcal{T}_{\text{out}}$ times, we apply the color transfer function to all pixels within $\mathcal{M}_0$.

5 Ethical, Legal, and Safety Consideration

The work on this project does not in and of itself entail many such issues. As this project is implemented in software and the work is done on a local machine, there are no safety implications in terms of physical health and cyber-security. Likewise, there are no ethical issues. From the legal perspective, both the source and candidate images must be properly cited to adhere to various copyright laws.

Although there are little such considerations when working on the project, the applications of image completion do raise concerns. From a legal point of view, a proprietary image which was completed using such algorithms, essentially making use of other images, must reference the exemplar image(s) used in the completion. Similarly, when discussing the ethical concerns around image completion, one quickly realizes that such algorithms may be used to construct visually plausible, yet misleading or derogatory images, which could be used with malicious intent.

6 Evaluation

As the verdict on the realism of an image is completely qualitative in nature, typical image completion evaluation schemes do not provide quantitative metrics derived from the quality of an image. Instead, experiments are performed on several images and are accompanied by a discussion of the image completion attempt's success. Surveys are also often distributed and the quantitative results from those are used as indications of the quality of a certain image completion technique in relation to others.

A similar approach is taken in this project. Upon completion of the technique, I will compare the results generated by it, as well as other works (provisionally, those by Huang and Dragotti [2], Zhu et al. [17], and Darabi et al. [8]). The images used for experiments will come from various sources, while fulfilling the requirement that for each source image to be completed there are at least ten semantically similar exemplar images. Several works have previously made use of the Oxford Building Dataset [36], which contains multiple images of various landmarks in Oxford, United Kingdom. Similarly, there is another dataset, Caltech-101, which contains images belonging to 101 categories. Despite providing a plethora of both source and exemplar images, such datasets often lack any capacities to investigate the effects of edge cases. Thus, I will also compile a database of source images, along with their candidate exemplar images using the Google search engine, designed specifically to examine the effects of edge cases. This will provide a more complete test for specific aspects of the image completion technique. In particular, in the set of candidate exemplar images for a particular occluded image, I will include images which are occluded in similar positions. This is aimed to test whether my implementation is able to pick the most similar images as exemplars (which in this case would be occluded as well and may not serve well for image completion) or the most "suitable" ones. In addition, I will include diverse images to be completed, in terms of the structures, patterns, and textures within the image, to observe how well my implementation performs for different inputs.

Finally, as in [2] and [14], a survey will be distributed to several people, where the results of various image completion techniques, including the one produced by this project, are presented and users are asked to rate the most realistic images, which will provide a metric for subjective evaluation.

7 Implementation Plan

This section details the key technical deliverables and timelines for this project, as well as fall-back options in the case that the proposed deliverables are not achievable in the alloted time. Figure 4 shows the detailed plans, where the numbers at the top represent weeks (week 1 represents 29/01/2018), and the tasks on the left are clarified in Table 1. The tasks in italics represent extensions that could be made given enough time but are not fundamental to the completion of the project. In particular, Task 7 is inspired by the concerns voiced in [18], where there is a fear that selecting the most similar image may actually select an exemplar that is occluded in the same region as the source image, consequently serving poorly for image completion. As a result, I will attempt to enhance the selection mechanism to account for images that are not occluded in the same region as the source. Although the provided timescales serve as an estimate for completion time, it is likely that an iterative approach will be employed, whereby certain deliverables will be revised even after their completion and some degree of testing will take place continuously to guide the development.

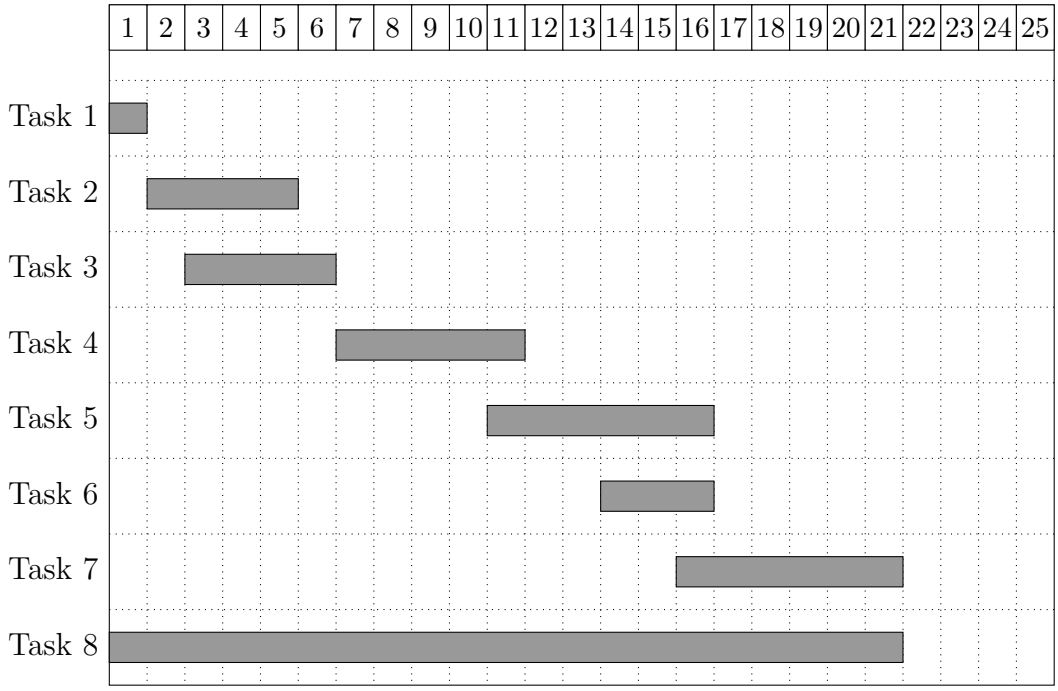

Figure 4: Gantt Chart showing timelines for various deliverables, where numbers at the top represent weeks starting from 29/1/2018, and the tasks are elaborated in Table 1

Table 1: Clarification of Tasks

Task	**Deliverable**	**Fall-back Deliverable**
1	Migrate software from Windows to OSX	Create a Windows VM for development purposes
2	Implement BRIEF-Gist descriptor and test performance	Implement standard BoW's and test performance
3	Compilation of testing database	None
4	Determine & implement most accurate image selection technique	None
5	Implement image completion using simultaneously NNF of both exemplars	Choose "best" exemplar from the two and use it for completion
6	Implement color correction to account for color discrepancies between the source and two exemplar images	Implement color correction to remedy color discrepancies between source and "best" exemplar
7	*Implement occlusion detection to select occlusion free images as exemplar*s	None
8	Final Report Write-Up	None

References

[1] Marcelo Bertalmio, Guillermo Sapiro, Vincent Caselles, and Coloma Ballester. Image inpainting. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 417–424, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co. ISBN 1-58113-208-5. doi: 10.1145/344779.344972. URL `http://dx.doi.org/10.1145/344779.344972`.

[2] Jun-Jie Huang and Pier-Luigi Dragotti. Photo-realistic image completion via dense correspondence. 2017.

[3] C. Guillemot and O. Le Meur. Image inpainting : Overview and recent advances. *IEEE Signal Processing Magazine*, 31(1):127–144, Jan 2014. ISSN 1053-5888. doi: 10.1109/MSP.2013.2273004.

[4] M. Bertalmio, L. Vese, G. Sapiro, and S. Osher. Simultaneous structure and texture image inpainting. *IEEE Transactions on Image Processing*, 12(8):882–889, Aug 2003. ISSN 1057-7149. doi: 10.1109/TIP.2003.815261.

[5] A. A. Efros and T. K. Leung. Texture synthesis by non-parametric sampling. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1033–1038 vol.2, 1999. doi: 10.1109/ICCV.1999.790383.

[6] A. Criminisi, P. Perez, and K. Toyama. Object removal by exemplar-based inpainting. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, volume 2, pages II–721–II–728 vol.2, June 2003. doi: 10.1109/CVPR.2003.1211538.

[7] C. W. Fang and J. J. J. Lien. Rapid image completion system using multiresolution patch-based directional and nondirectional approaches. *IEEE Transactions on Image Processing*, 18(12):2769–2779, Dec 2009. ISSN 1057-7149. doi: 10.1109/TIP.2009.2027635.

[8] Soheil Darabi, Eli Shechtman, Connelly Barnes, Dan B. Goldman, and Pradeep Sen. Image melding: Combining inconsistent images

using patch-based synthesis. *ACM Trans. Graph.*, 31(4):82:1–82:10, July 2012. ISSN 0730-0301. doi: 10.1145/2185520.2185578. URL http://doi.acm.org/10.1145/2185520.2185578.

[9] Z. Xu and J. Sun. Image inpainting by patch propagation using patch sparsity. *IEEE Transactions on Image Processing*, 19(5):1153–1165, May 2010. ISSN 1057-7149. doi: 10.1109/TIP.2010.2042098.

[10] K. He and J. Sun. Image completion approaches using the statistics of similar patches. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(12):2423–2435, Dec 2014. ISSN 0162-8828. doi: 10.1109/TPAMI.2014.2330611.

[11] M. Köppel, M. Ben Makhlouf, K. Müller, and T. Wiegand. Fast image completion method using patch offset statistics. In *2015 IEEE International Conference on Image Processing (ICIP)*, pages 1795–1799, Sept 2015. doi: 10.1109/ICIP.2015.7351110.

[12] N. Komodakis. Image completion using global optimization. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 1, pages 442–452, June 2006. doi: 10.1109/CVPR.2006.141.

[13] Y. Pritch, E. Kav-Venaki, and S. Peleg. Shift-map image editing. In *2009 IEEE 12th International Conference on Computer Vision*, pages 151–158, Sept 2009. doi: 10.1109/ICCV.2009.5459159.

[14] James Hays and Alexei A. Efros. Scene completion using millions of photographs. *ACM Trans. Graph.*, 26(3), July 2007. ISSN 0730-0301. doi: 10.1145/1276377.1276382. URL http://doi.acm.org/10.1145/1276377.1276382.

[15] Oliver Whyte, Josef Sivic, and Andrew Zisserman. Get out of my picture! internet-based inpainting. In *BMVC*, 2009.

[16] Hanieh Amirshahi, Satoshi Kondo, Koichi Ito, and Takafumi Aoki. An image completion algorithm using occlusion-free images from internet photo sharing sites. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, E91-A(10):2918–2927, October 2008. ISSN 0916-8508. doi: 10.1093/ietfec/e91-a.10.2918. URL http://dx.doi.org/10.1093/ietfec/e91-a.10.2918.

[17] Z. Zhu, H. Z. Huang, Z. P. Tan, K. Xu, and S. M. Hu. Faithful completion of images of scenic landmarks using internet images. *IEEE Transactions on Visualization and Computer Graphics*, 22(8):1945–1958, Aug 2016. ISSN 1077-2626. doi: 10.1109/TVCG.2015.2480081.

[18] A. Wong and J. Orchard. A nonlocal-means approach to exemplar-based inpainting. In *2008 15th IEEE International Conference on Image Processing*, pages 2600–2603, Oct 2008. doi: 10.1109/ICIP.2008.4712326.

[19] Aude Oliva and Antonio Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175, May 2001. ISSN 1573-1405. doi: 10.1023/A:1011139631724. URL `https://doi.org/10.1023/A:1011139631724`.

[20] R. Talat, M. Muzammal, and I. Siddiqi. Scene completion using top-1 similar image. In *2016 International Conference on Frontiers of Information Technology (FIT)*, pages 252–257, Dec 2016. doi: 10.1109/FIT.2016.053.

[21] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, Nov 2004. ISSN 1573-1405. doi: 10.1023/B:VISI.0000029664.99615.94. URL `https://doi.org/10.1023/B:VISI.0000029664.99615.94`.

[22] Ting-Zhu Dai, Chang-Sheng Tan, and Qing Liu. On content-based image synthesis. In *2012 8th International Conference on Computing Technology and Information Management (NCM and ICNIT)*, volume 2, pages 607–612, April 2012.

[23] J. Sivic and A. Zisserman. Video google: a text retrieval approach to object matching in videos. In *Proceedings Ninth IEEE International Conference on Computer Vision*, pages 1470–1477 vol.2, Oct 2003. doi: 10.1109/ICCV.2003.1238663.

[24] Y. Chen, A. Dick, and X. Li. Visual distance measures for object retrieval. In *2012 International Conference on Digital Image Computing Techniques and Applications (DICTA)*, pages 1–8, Dec 2012. doi: 10.1109/DICTA.2012.6411668.

[25] S. Belongie, C. Carson, H. Greenspan, and J. Malik. Color- and texture-based image segmentation using em and its application to content-based image retrieval. In *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*, pages 675–682, Jan 1998. doi: 10.1109/ICCV.1998.710790.

[26] N. Sünderhauf and P. Protzel. Brief-gist - closing the loop by simple means. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1234–1241, Sept 2011. doi: 10.1109/IROS.2011.6094921.

[27] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 110(3):346 – 359, 2008. ISSN 1077-3142. doi: https://doi.org/10.1016/j.cviu.2007.09.014. URL `http://www.sciencedirect.com/science/article/pii/S1077314207001555`. Similarity Matching in Computer Vision and Multimedia.

[28] Y. Jeon, T. Lee, C. Kim, D. Yi, and D. I. D. Cho. A brief-gist based efficient place recognition for indoor home service robots. In *2016 16th International Conference on Control, Automation and Systems (ICCAS)*, pages 1526–1530, Oct 2016. doi: 10.1109/ICCAS.2016.7832506.

[29] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 28(3), August 2009.

[30] Connelly Barnes, Eli Shechtman, Dan B Goldman, and Adam Finkelstein. The generalized PatchMatch correspondence algorithm. In *European Conference on Computer Vision*, September 2010.

[31] Christian Bailer, Bertram Taetz, and Didier Stricker. Flow fields: Dense correspondence fields for highly accurate large displacement optical flow estimation. *CoRR*, abs/1508.05151, 2015. URL `http://arxiv.org/abs/1508.05151`.

[32] M. Calonder, V. Lepetit, M. Ozuysal, T. Trzcinski, C. Strecha, and P. Fua. Brief: Computing a local binary descriptor very fast. *IEEE*

Transactions on Pattern Analysis and Machine Intelligence, 34(7):1281–1298, July 2012. ISSN 0162-8828. doi: 10.1109/TPAMI.2011.222.

[33] C. A. Micchelli and M. Pontil. On learning vector-valued functions. *Neural Computation*, 17(1):177–204, Jan 2005. ISSN 0899-7667. doi: 10.1162/0899766052530802.

[34] Jiayi Ma, Ji Zhao, Jinwen Tian, Xiang Bai, and Zhuowen Tu. Regularized vector field learning with sparse approximation for mismatch removal. *Pattern Recognition*, 46(12):3519 – 3532, 2013. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2013.05.017. URL `http://www.sciencedirect.com/science/article/pii/S0031320313002410`.

[35] Gianluca Donato and Serge Belongie. Approximate thin plate spline mappings. 07 2001.

[36] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007.