2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

Structure-from-Motion Reconstruction Based on Weighted Hamming Descriptors

Guoyu Lu, Vincent Ly and Chandra Kambhamettu
Video/Image Modeling and Synthesis Lab
University of Delaware
Newark, DE, USA

***Abstract*—We propose a pipelined methods to reduce memory consumption of large-scale Structure-from-Motion reconstruction with the use of unsorted images extracted from photo collection websites. Recent research is able to reconstruct cities based on extracted images from photo collection websites. SIFT feature is used to find the correspondences between two images. For the large-scale reconstruction with unsorted images, the system needs to store all the descriptors and feature points information in memory to search for correspondences. As each SIFT descriptor is a 128 dimensional real-value vector, storing all the descriptors would consume a significant amount of memory. Based on this limitation, we project the high dimensional features into a low-dimensional space using a learned projection matrix. After projection, the distance of the descriptors belonging to the same point in 3D space is decreased; the distance of the descriptors belonging to the different points is increased. Furthermore, we learn a mapping function, which maps the real-value descriptor into binary code. As Hamming descriptors contain only two value options per bit and the length of the descriptor is limited, there are usually multiple descriptors having the same Hamming distance to the query descriptor. In dealing with this problem, we give different weights to each dimension and rank each bit of the Hamming descriptor based on each dimensions discriminant power; this contributes to reduce the ambiguity in matching the descriptors. The experiments show that our method achieves dense reconstruction results with less than 10 percent of the original memory consumption.**

I. INTRODUCTION

3D reconstruction has large usage in many areas, such as computer games, virtual realities, and movie industries. There are mainly two kinds of image-based 3D reconstruction techniques: stereo reconstruction and Structure-from-Motion reconstruction (SfM). Compared with stereo reconstruction, Structure-from-Motion reconstruction does not require the cameras to be calibrated while capturing images. For this reason, Structure-from-Motion reconstruction is suitable for large-scale reconstruction tasks because the images are easier to obtain. With an increase in smartphone usage, an increasing number of images are uploaded to photo collection websites (e.g. Flicker). This provides the possibility of collecting a large number of unordered images, typically between several hundred or several thousand, for reconstructing a large building structure. SfM analyzes the images and builds a 3D model based on matching interesting points among the images. Agarwal et al. [1] reconstruct a scene based on a large number of images from a photo collection website. To reconstruct a multitude of images efficiently, the system separates the image feature matching process to multiple stages, each of which contains a proposal and verification step. The proposal step determines the set of images. The verification step performs the feature matching process among the image pairs. Images passing the verification step are passed to the next step. The whole system is based on parallel computing to accelerate the whole reconstruction process.

SIFT feature is applied to detect the correspondences of the images due to the property of invariance to rotation, transformation, scaling, and illumination changes. On an image with rich textures, usually at least several hundred SIFT features are detected. Each SIFT feature contains 128 dimension real-value numbers, consuming a great amount of memory to store the features. In order to deal with this large memory consumption, we learn a projection matrix that projects the high dimensional descriptors to a low dimensional space. The lower dimensional real-value descriptors are mapped to the binary space to further reduce the memory cost. With lower memory consumption, the distance calculation is simplified. The projected lower dimensional real-value descriptors are projected to Hamming space. After projecting to Hamming space, multiple Hamming descriptors may share the same distance to the query descriptor. We solve the distance confusion in Hamming space by giving different weights to each dimension based on the discriminant power of each dimension. Our experiments show that our weighted Hamming descriptor achieves good result with less than 10 percent of the original memory.

Section II briefly introduces the related work. Section III presents our method for learning the projection matrix and the mapping method to learn Hamming descriptors. Section IV introduces our binary bit weighting method in Hamming space. Section V gives an introduction to the 3D reconstruction pipeline. Section VI discusses the experiment result, and section VII concludes the paper.

II. RELATED WORK

Stereo and SfM reconstruction are two dominant 3D reconstruction methods based on images. SfM reconstruction does not restrict cameras to be calibrated and thus used in many large-scale reconstruction with easily available images, unlike stereo reconstruction. Hartley et al. [2] introduce the reconstruction methods based on projective geometry and 3D reconstruction from un-calibrated cameras. Fundamental matrix based approach [3] and factorization based approach

978-1-4799-1484-5/14/$31.00 ©2014 IEEE

[4] form the two main ways for projective reconstruction. Fruh et al. [5] capture the 2D images and the corresponding depth information through a video camera and two laser scanners mounted on a truck. They reconstruct a city model by the 2D images and the depth information calculated by the laser scanners. Pollefeys et al. [6] realize a real-time city reconstruction system for video streams based on GPS and inertia technologies. Zebedin et al. [7] combine aerial images, sparse line features delineating height discontinuities, and dense depth data to build the dense city reconstruction model. All of these methods are reliant on calibrated cameras or laser scanners. Smartphone and internet technologies development significantly increases the number of images on the Internet. Snavely et al. [8], [9] extract Internet images to reconstruct a structure based on SfM reconstruction method. Agarwal et al. [1] build a large-scale reconstruction system using images from photo collection websites; the system contains several stages, each of which is composed by a proposal and verification step to build the feature correspondences among image pairs. The system is employed on a computer cloud to accelerate the reconstruction procedure. Instead of utilizing a computer cloud, Frahm et al. [10] obtain a highly parallel implementation on graphics processors and multi-core architectures with the application of geometry and appearance constraints. Lu et al. [11] conduct the SfM reconstruction based on reduced dimensional descriptors and learned Hamming descriptors. However, the confusion may happen when several Hamming descriptors have the same distance to the query descriptor while selecting the best matching candidate.

Many high level descriptors contain high dimensionality, such as 128-dimension SIFT feature [12] and 64-dimensions SURF feature [13], [14]. The high feature dimensionality would not be a big concern for dealing with small number of features. However, for large-scale reconstruction problems, the high dimensionality would consume a large portion of the memory. Research has been invested to reduce the high dimensional feature to the low dimensional space. Based on Principal Component Analysis [15], Yan et al. [16] develop PCA-SIFT, which reduces the SIFT feature dimensionality. Hua et al. [17] adopt a discriminative approach from labeled match and non-match pairs to learn a lower dimensional embedding. Making use of Linear Discriminates Analysis [18] and Powell minimization [19] , Brown et al. [20] reduce the descriptors' dimensionality through both linear and nonlinear transforms. Philbin et al. [21] learn a projection matrix by minimizing a margin-based cost function built on the basis of three groups of descriptor pairs, which are positive pairs, nearest neighbor negative pairs, and random negative pairs.

Distance computation in binary code is much easier than the real-value numbers and the descriptors storing in Hamming space could save a great amount of space. Kulis et al. [22] generate locality-sensitive hashing for arbitrary kernel functions and prove their usage in indexing local patch descriptors. Spectral Hashing was proposed by Weiss et al. [23] to compute binary code for a new data point by selecting a subset of eigenvectors of graph Laplacian. Yagnik et al.[24] present an embedding method (WTAHash) based on partial order statistics. Their embedding method exhibits the simplicity in implementation and effectiveness in distance computation. The special case of their Hashing method when applied to binary vectors is the well-known MinHash [25], [26] method.

III. Projection matrix and mapping function learning

A. Projection matrix learning

The main reason for the large memory requirement is the high dimensionality of the SIFT feature. To deal with this problem, we learn a projection matrix that projects the high dimensional SIFT features to the low dimension domain. Ideally, after projection, the distance between positive descriptors is reduced and the distance between negative descriptors is enlarged. The positive descriptors represent the descriptors from the same point in 3D point cloud. The positive descriptors in different images should fit RANSAC transformation [27] and negative descriptors mean that the descriptors are from the different points. As a result, the descriptors of different points cannot find the transformation matrix while running the RANSAC. We learn the projection matrix based on the positive and negative descriptor pairs. As mentioned above, positive descriptors are strictly defined as the matching descriptors satisfying the RANSAC transformation. Negative descriptors are mainly the descriptor pairs that match each other, but not satisfy the RANSAC transformation. However, we want to keep the negative descriptor pairs the same number as the positive descriptors, and normally matching descriptors that do not fit RANSAC are less than the matching descriptors satisfying the RANSAC transformation. To take this difference into consideration, we randomly select a group of non-matching descriptors as the negative descriptor pairs. The positive descriptors' distance and the negative descriptors' distance are represented by Eq.1 and Eq.2.

$$D_{Pos} = \sum_{x,x' \in Pos} (Wx - Wx')^2 \tag{1}$$

$$D_{Neg} = \sum_{x,x' \in Neg} (Wx - Wx')^2 \tag{2}$$

In the above equations, W represents the projection matrix. x, x' are the original descriptors. Pos and Neg are separately meaning the positive and negative descriptor pairs. We further present the distance equation in terms of Trace function:

$$D_{Pos} = \sum_{x,x' \in Pos} (Wx - Wx')^2 = Tr(W^T(\sum_{Pos}(x - x')^2)W) \tag{3}$$

$$D_{Neg} \sum_{x,x' \in Neg} (Wx - Wx')^2 = Tr(W^T(\sum_{Neg}(x - x')^2)W) \tag{4}$$

To learn the projection matrix, we build a cost function in terms of increasing the negative descriptor pairs' distance and

decreasing positive descriptor pairs' distance, as shown in Eq. 5.

$$cost = max \frac{Tr(W^T(\sum_{Neg}(x-x')^2)W)}{Tr(W^T(\sum_{Pos}(x-x')^2)W)} \tag{5}$$

[28] points out that minimizing Eq.6 is the same as calculating the root of equation $f(\eta) = 0$. Here, $f(\eta)$ is described in Eq. 7.

$$max \frac{Tr(W^T S_m W)}{Tr(W^T S_n W)} \tag{6}$$

$$f(\eta) = maxTr(W^T(S_m - \eta S_n)W) \tag{7}$$

S_m and S_n are two matrices.

$$\eta = \frac{Tr(W^T S_m W)}{Tr(W^T S_n W)} \tag{8}$$

Similarly, we choose a value of λ as the Eq. 9 to optimize projection matrix W for minimizing the $Cost$ of Eq. 5.

$$\lambda = \frac{Tr(W^T(\sum_{Pos}(x-x')^2)W)}{Tr(W^T(\sum_{Neg}(x-x')^2)W)} \tag{9}$$

We initialize the projection matrix W as an arbitrary orthogonal matrix, which satisfies $W^T W = I$. And W is optimized by performing the eigen-decomposition of $(\sum_{Pos}(x-x')^2 - \lambda \sum_{Neg}(x-x')^2)$ to change the value of λ. This process is repeated until W converges. To increase the convergence speed, Yang et al. [29] propose to substitute the eigen-decompostion of $(\sum_{Pos}(x-x')^2 - \lambda \sum_{Neg}(x-x')^2)$ by the following steps.

1) Calculate the d eigenvectors $(e1, e2, ..., ed)$ of $(\sum_{Pos}(x-x')^2 - \lambda \sum_{Neg}(x-x')^2)$. Repeat the following two operations until W does not change any more.

2) Sort $ei^T(\sum_{Pos}(x-x')^2 - \lambda \sum_{Neg}(x-x')^2)ei, i = 1, 2...d$ in descending order and select the first d' eigenvectors to construct W. d' is the final dimension of the descriptors.

3) Compute $\lambda = \frac{Tr(W^T(\sum_{Pos}(x-x')^2)W)}{Tr(W^T(\sum_{Neg}(x-x')^2)W)}$.

After the above the steps converging, W is the projection matrix that projects the original descriptor to d' dimensions.

B. Hamming descriptor mapping

After projecting the original high dimensional descriptors to low dimensional space, we map the low dimensional descriptors to Hamming space. We adopt the hashing method proposed by Strecha et al. [30] to further map the lower dimensional descriptors to Hamming space. It is simpler and faster to compute the distance in Hamming space than Euclidean space. Since one dimension in Hamming space takes only one bit, the memory cost is greatly reduced by mapping the descriptor from Euclidean space to Hamming space. Nonetheless, an integer value requires 8 bits and the float spends 16 bits in most programming languages. The mapping from real-value descriptor to Hamming descriptor is through the following operation:

$$y = sign(x - T) \tag{10}$$

x is the real-value descriptor. y is the corresponding value in Hamming space. T represents the threshold. If $(x - T)$ is larger than 0, y is assigned to be 1 by the $sign$ operation. On the other hand, if $(x - T)$ is smaller than 0, $sign$ operation assigns -1 to y. Thus, the Hamming descriptor is achieved by comparing the real-value descriptor with a threshold.

The most essential step in mapping a real-value number to a binary descriptor is to learn the threshold. The learned threshold must minimize the false matching rate, composed by the false positive rate and the false negative rate. The false positive rate is calculated by the following equation:

$$\begin{aligned} FP(T) =& Pr\{\min(x_{Neg}, y_{Neg}) \geq T \\ & \cup \max(x_{Neg}, y_{Neg}) < T) | Neg\} \\ =& 1 - cdf(\min(x_{Neg}, y_{Neg}) | Neg) \\ & + cdf(\max(x_{Neg}, y_{Neg}) | Neg) \end{aligned} \tag{11}$$

where x and y are two descriptors after projection at this situation. Neg represents the negative descriptor pairs, which are the resource of the descriptor pair (x, y); T is the threshold; cdf is the cumulative distribution function for each T value. Similarly, the false negative rate is computed by Eq.12.

$$\begin{aligned} FN(T) =& Pr\{\min(x_{Pos}, y_{Pos}) < T \leq \max(x_P, y_P) | Pos\} \\ =& Pr\{(\min(x_{Pos}, y_{Pos} < T) | Pos\} + 1 \\ & - Pr\{(\max(x_{Pos}, y_{Pos}) < T) | Pos\} \\ =& cdf\{\min(x_{Pos}, y_{Pos}) | Pos\} \\ & - cdf\{\max(x_{Pos}, y_{Pos}) | Pos\} \end{aligned} \tag{12}$$

The total false rate is the sum of the false positive rate and the false negative rate, as Eq.13.

$$F(T) = FP(T) + FN(T) \tag{13}$$

The threshold value which supplies the smallest F is the threshold we use to map the real-value descriptors to Hamming descriptors.

IV. Bit weighting in Hamming space

After storing the descriptors in Hamming space, the storage consumption can be largely reduced while the distance computation between descriptor pairs is also getting much easier. With the benefit of efficient storage and easy computation, a problem of Hamming descriptor arises that the distance between every descriptor pair is limited to a certain integer range since the descriptor length is limited and each dimension of the descriptor has only two value options, either 0 or 1. Due to this limitation, usually more than one descriptors share the same distance to the query descriptor, resulting in a relatively

high false positive descriptor selection rate by randomly selecting one descriptor among all nearest neighbors of the query descriptor. To deal with this problem, [31] conducts a majority vote based on the associated 3D points of the 10 nearest neighbors to find the correspondence between 2D features and 3D points, making use of the fact that every 3D point has multiple descriptors associated. Different from the localization problem in [31], we are trying to find the correspondences between 2D images, where every point has only one descriptor associated. In our paper, this problem is addressed by giving a weight to each dimension of the Hamming descriptor and ranking the weighted bit. The weight is given based on the learned discriminant power of the dimension, which helps to reduce the distance ambiguity among all the matching descriptors.

While learning the Hamming descriptors, we aim to reduce the distance of the positive descriptors in Hamming space as much as possible. In the best situation, the Hamming descriptors coming from the positive pairs would have exactly the same value in each dimension. However, this situation does not happen in most cases. Usually, there are several bits differ for the positive descriptors. The value of each dimension distributes differently among all of the descriptors. In some dimensions, the different values may happen frequently while the difference occurs rarely in some other dimensions. This phenomenon results in the different discriminant powers among all the dimensions of the learned descriptors. For instance, two Hamming descriptors, $Desc_1$ and $Desc_2$, both differ 1 bit to the query descriptor $Desc_3$. $Desc_1$ is different from $Desc_3$ on the m th bit, while $Desc_2$ contains a different value from $Desc_3$ on the n th dimension. Assuming the m th dimension has a power more discriminant than the n th dimension, the distance between $Desc_1$ and $Desc_3$ is considered larger than the distance between $Desc_2$ and $Desc_3$.

We learn the discriminant power with the purpose of decreasing the positive descriptors distance and increasing the negative descriptors distance. The real-value descriptors after projection are used in learning the weight of each dimension of the descriptor due to the large information loss for Hamming descriptors. Before hashing, on a certain dimension, if the values of positive descriptors is quite close to each other and very distant to other negative descriptors, this dimension retains a high discriminant power and the derived Hamming bit on this dimension is quite likely to generate an accurate distance result; on the other hand, if the positive and negative descriptors' values cannot be really distinguished, we do not hold high confidence on this dimension for getting the correct Hamming bit. For the k th dimension, the average distance of all the positive descriptor pairs is shown as Eq. 14.

$$DP_k = \frac{\sum_{(x_k,y_k)\in Pos}\sqrt{(x_k-y_k)^2}}{Npos} \tag{14}$$

In Eq. 14, $Npos$ is the number of positive descriptor pairs. x_k and y_k are the values of the k th dimension of a positive descriptor pair. Similarly, the average distance of negative descriptor pairs is described by Eq. 15,

$$DN_k = \frac{\sum_{(x'_k,y'_k)\in Neg}\sqrt{(x'_k-y'_k)^2}}{Nneg} \tag{15}$$

where $Nneg$ is the number of negative descriptor pairs. x'_k and y'_k are the values of the k th dimension of the descriptors from negative pairs. The weight of the k th dimension is generated by the scale between DN_k and DP_k, shown as Eq. 16.

$$\begin{aligned} W_k =& \frac{DN_k}{DP_k} = \frac{\frac{\sum_{(x'_k,y'_k)\in Neg}\sqrt{(x'_k-y'_k)^2}}{Nneg}}{\frac{\sum_{(x_k,y_k)\in Pos}\sqrt{(x_k-y_k)^2}}{Npos}} \\ =& \frac{Npos*(\sum_{(x'_k,y'_k)\in Neg}\sqrt{(x'_k-y'_k)^2})}{Nneg*(\sum_{(x_k,y_k)\in Pos}\sqrt{(x_k-y_k)^2})} \end{aligned} \tag{16}$$

We sum up all the descriptor pairs' Euclidean distance and further divide the distance sum by the negative descriptor pairs number. The same for positive descriptors to get the average negative descriptor distance on the k th dimension. The scale between the average negative descriptor distance and the average positive descriptor distance is assigned to this dimension as the weight. The time complexity in computing the positive and negative descriptor pairs' distance is $O(n^2)$. For the large scale structure from motion reconstruction task, learning the Hamming descriptor dimensions' weights would consume a large amount of time. To address this issue, we use the positive descriptor pairs' standard deviation to substitute the Euclidean distance in order to get the average positive descriptor distance. And the standard deviation of all the descriptors is used as the negative descriptors as the following equations.

$$DN' = \sqrt{\frac{1}{N-1}\sum_1^N (y_k-\mu)^2} \tag{17}$$

$$DP' = \frac{\sum_1^{Np}\sqrt{\frac{1}{NDes}\sum_1^{NDes}(x_k-\mu_{des})}}{Np} \tag{18}$$

In the equations above, we represent N as the number of all the descriptors, and μ is the mean value of all the descriptors' k th dimension value. For the positive standard deviation part, Np describes the number of different points. $Ndes$ is the descriptor number of the current point. μ_{des} represents the k th dimension's mean value of the positive descriptors. Eq.18 and Eq. 17 provide the distribution of the descriptor's value before hashing. The new distance computed by standard deviation is applied in calculating the weight of the k the dimesnion, shown as Eq. 19.

$$W_k = \frac{DN'}{DP'} = \frac{\sqrt{\frac{1}{N-1}\sum_1^N (y_k-\mu)^2}}{\frac{\sum_1^{Np}\sqrt{\frac{1}{NDes}\sum_1^{NDes}(x_k-\mu_{des})}}{Np}} = \frac{Np*(\sqrt{\frac{1}{N-1}\sum_1^N (y_k-\mu)^2})}{\sum_1^{Np}\sqrt{\frac{1}{NDes}\sum_1^{NDes}(x_k-\mu_{des})}} \quad (19)$$

Following the idea of Eq. 16, the weight of k th dimension is generated by the division of average negative descriptor pairs distance DN' and average positive descriptor pairs distance DP'. The larger negative descriptor value distance and the smaller positive descriptor value distance, the more discriminant power this dimension has and the more correctness confidence we have for this dimension. Thus, we give a higher weight to this dimension. As Eq.19 makes use of the standard deviation, we get rid of computing the Euclidean distance of each descriptor pair. In this way, the time complexity is $O(n)$. This could extensively reduce the time consumption while computing the dimension weights of large datasets compared with the method used in Eq. 16. The sum of the weights of the corresponding dimensions that differ from each other is the distance between these two Hamming descriptors.

V. Reconstruction pipeline

The basic reconstruction pipeline introduced by Agarwal et al. [1] is adopted in our reconstruction pipeline. The first step of the reconstruction is extracting features and performing matching. SIFT is applied as the local feature to perform matching. To reduce search cost, Approximate Nearest Neighbor (ANN)[32] is used efficiently to determine the matching point. Features of one image are inserted into a k-d tree and features in other images are used as the queries. In the construction of the vocabulary searching tree, hierarchical k-means is used to quantize the features in an image. The quantization is aggregated over all features in an image to obtain a term frequency (TF) vector for the image and a document frequency (DF) for the corpus of the images. SIFT matching ratio is also used in the pipeline to retain the accurate matching number, where the distance to the nearest neighbor should be twice smaller than the one to the second nearest neighbor.

RANSAC [27] is used to cull the outliers to enhance the accuracy of the matches. Agarwal et al. [1] applied a computer cloud to conduct the reconstruction task. However, our main goal is to reduce the memory consumption so that we use a single computer instead of a computer cloud to accomplish the reconstruction; we do not deal with the additional complication of retrieving images from various computers. The top $k1$ images with the highest TFIDF score are further utilized for verification; meanwhile, another $k2$ images with the highest TFIDF score after $k1$ are used to enrich the connected components. The essential matrix between the image pair is estimated to verify image pairs. Once the image pair meets the following three conditions, a full Euclidean two-view reconstruction is produced and stored in the memory. The 3 conditions are: 1) Essential matrix estimation is successful. 2) A sufficient angle exists between the directions of two cameras. 3) There are enough matching points.

We further merge the connected components, where $k2$ images in the proposal play the role of connection. After obtaining the Euclidean two-view reconstruction by $k1$ images with the highest TFIDF score, the $k2$ images connect the two-view reconstructions. Any two images in $k1$ matching with the same image in $k2$ are connected. Hence, images in $k2$ that do not match any image in $k1$ are discarded. A matching graph could be practically built after the above steps. However, this matching graph is usually not dense enough to build a good reconstruction model. In order to make the model more dense, the matched images create an expansion for seeking more matched images. For instance, image i is connected to image j and image j is connected to image k. We then check whether image i is also connected to image k; this process should be stopped after a certain number of iterations. Two images are connected if there are enough matching features. After connecting the entire image matching tracks, the matching process is complete.

The last step is to perform the Structure-from-Motion reconstruction (SfM). Some photos from the Internet contain important parts of the buildings with characteristic texures, while some other photos consist of only small portion of the buildings. Due to this difference, it is preferable to first reconstruct a small image set that has essential image connectivity; this connectivity is built through searching large loops within the image connections. After the small essential image set is reconstructed, we incrementally reconstruct all the remaining images using pose estimation and triangulate all the remaining points. Bundle adjustment [33] is applied to minimize the reconstruction error of the reconstructed model.

VI. Experiments

We use Notre Dame dataset in our experiments, which contains 715 images extracted from Flickr. The photos capture the different views of Notre Dame Cathedral. Due to the usage of a single computer instead of a computer cloud, we randomly select 50 images to do the reconstruction for verifying our method. Some of our images used for reconstruction are shown in Fig. 1.

As shown in Fig 1, images contain large rotation, translation, scaling, illumination change and occlusion. We first give the reconstruction result using the original SIFT descriptor describes in Fig. 2. There are 16895 points reconstructed with the utilization of the original SIFT descriptor. A point cloud of the gate of Notre Dame Cathedral is observed from Fig 2.

For Hamming descriptor, as indicated by [31] , 96 dimensional Hamming descriptor usually provides the best trade-off between the memory consumption and accuracy so that we choose 96 dimensional Hamming descriptor for reconstruction. The reconstruction result is given on Fig.3.

The reconstruction model contains 2656 reconstructed points. Though the reconstruction model is much sparser

Fig. 1. Images randomly selected for SFM reconstruction

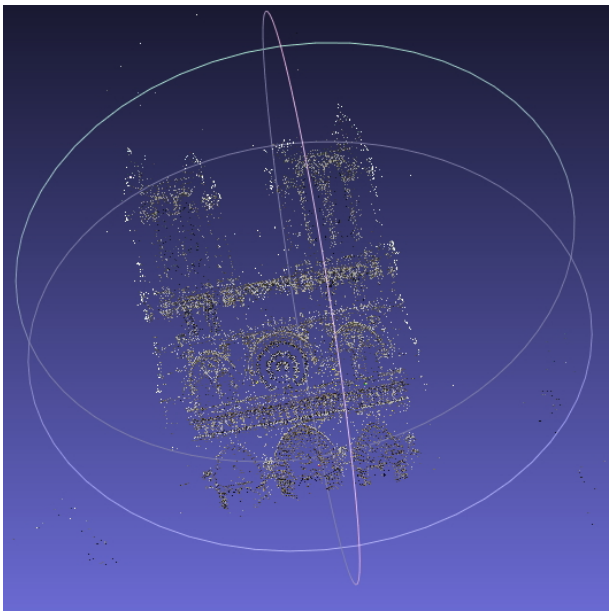

Fig. 2. Reconstruction model using original SIFT descriptor. 16895 points are reconstructed.

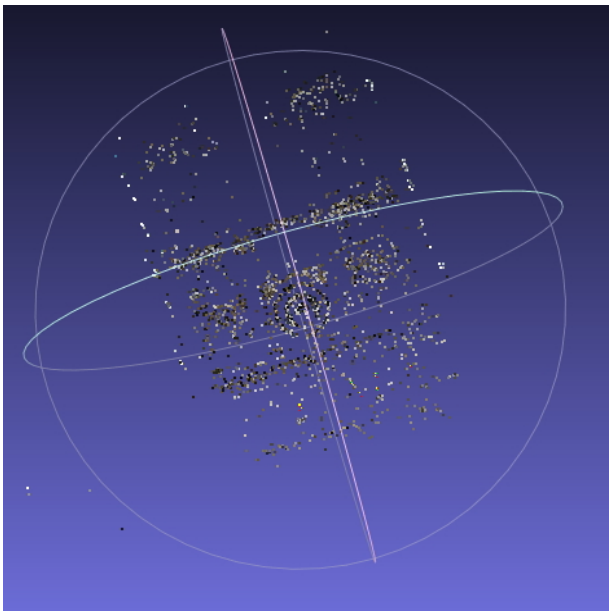

Fig. 3. Recontruction model using Hamming descriptor. There are 2656 points in this reconstruction model.

than the one using the original SIFT descriptor, the memory consumption is just $(1/8) * (96/128) = 9.38\%$, which is less than 10 percent of the original memory consumption. After adding the weight to each dimension of the descriptor, the reconstruction result is much denser, as shown in Fig. 4.

After adding weights to different dimensions of the descriptor, the matching confusion is greatly avoided. There are 8765 points reconstructed, which is a considerable improvement based on the simple Hamming descriptor with the same memory consumption. From the experiments, we can see that

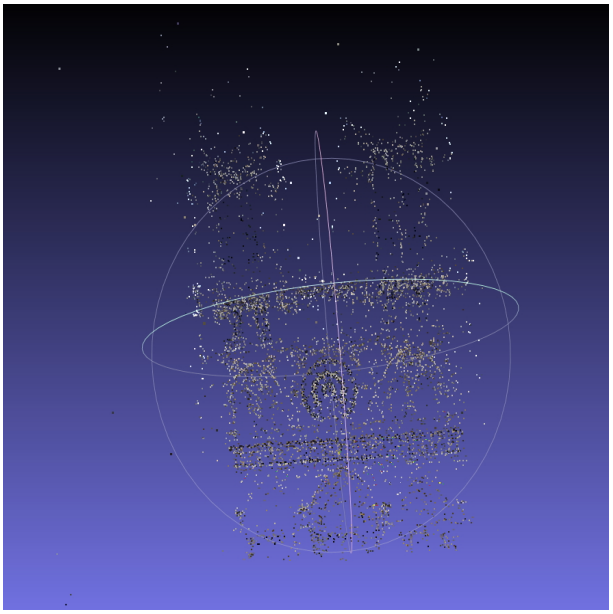

Fig. 4. Recontruction model using weighted Hamming descriptor. 8765 points are reconstructed.

our weighted Hamming descriptors can perform very well on reconstruction while saving large memory cost.

VII. Conclusion

In this paper, we present our method of Structure-from-Motion reconstruction with the use of weighted Hamming descriptors. The learned Hamming descriptors saves a significant portion of space to store the descriptors by hashing real-value descriptors to Hamming space. However, the descriptors after hashing may result in the confusion of selecting best matching candidate due to the descriptors information loss. As a result, multiple descriptors have the same Hamming distance to the query descriptor. To solve this problem, we learn the discriminant power of each descriptor dimension based on the projected real-value descriptors. According to the discriminant power we learned, different weights are given to each dimension of the Hamming descriptors. And the importance of the different dimension Hamming bits is ranked based on the associated weigths. This reduces the confusion of finding correspondences when multiple Hamming descriptors having the same distance to the query descriptors. Experiments show that our methods can give dense reconstruction result while largely reducing the memory consumption.

ACKNOWLEDGMENTS

The authors are grateful for the discussion with Dr. Yi Yang from Queensland University. We would like to thank to Dr. Noah Snavely from Cornell University for providing the Bundler code and the Notre Dame dataset.

References

[1] S. Agarwal, N. Snavely, I. Simon, S. Seitz, and R. Szeliski, "Building rome in a day," in *ICCV*, 2009, pp. 72–79.

[2] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*, vol. 2, Cambridge Univ Press, 2000.

[3] O. Faugeras, Q.T. Luong, and T. Papadopoulo, *The geometry of multiple images: the laws that govern the formation of multiple images of a scene and some of their applications*, the MIT Press, 2004.

[4] P. Sturm and B. Triggs, "A factorization based algorithm for multi-image projective structure and motion," in *ECCV*, 1996.

[5] C. Fruh and A. Zakhor, "An automated method for large-scale, ground-based city model acquisition," *International Journal of Computer Vision*, vol. 60, no. 1, pp. 5–24, 2004.

[6] M. Pollefeys, D. Nister, J.M. Frahm, A. Akbarzadeh, P. Mordohai, B. Clipp, C. Engels, D. Gallup, S.J. Kim, P. Merrell, et al., "Detailed real-time urban 3d reconstruction from video," *International Journal of Computer Vision*, vol. 78, no. 2-3, pp. 143–167, 2008.

[7] L. Zebedin, J. Bauer, K. Karner, and H. Bischof, "Fusion of feature- and area-based information for urban buildings modeling from aerial imagery," in *ECCV*. 2008.

[8] N. Snavely, S. M. Seitz, and R. Szeliski, "Photo tourism: exploring photo collections in 3d," in *ACM transactions on graphics (TOG)*, 2006, vol. 25, pp. 835–846.

[9] N. Snavely, S. Seitz, and R. Szeliski, "Skeletal graphs for efficient structure from motion.," in *CVPR*, 2008, vol. 1, p. 2.

[10] J.M. Frahm, P. Fite-Georgel, D. Gallup, T. Johnson, R. Raguram, C. Wu, Y.H. Jen, E. Dunn, B. Clipp, S. Lazebnik, et al., "Building rome on a cloudless day," in *ECCV*. 2010.

[11] G. Lu, V. Ly, and C. Kambhamettu, "Large-scale structure-from-motion reconstruction with small memory consumption," in *The 11th International Conference on Advances in Mobile Computing and Multimedia*. 2013.

[12] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[13] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[14] R. Kalia, Lee K., S. B.V.R., S.K. Je, and Oh W.G., "An analysis of the effect of different image preprocessing techniques on the performance of surf: Speeded up robust features," in *2011 17th Korea-Japan Joint Workshop on Frontiers of Computer Vision*, 2011, pp. 1–6.

[15] I.T. Jolliffe, *Principal Component Analysis*, Springer Verlag, 1986.

[16] Y. Ke and R. Sukthankar, "PCA-SIFT: a more distinctive representation for local image descriptors," in *CVPR*, 2004, vol. 2, pp. 506–513.

[17] G. Hua, M. Brown, and S. Winder, "Discriminant embedding for local image descriptors," in *ICCV*, 2007, pp. 1–8.

[18] S. Mika, G. Ratsch, J. Weston, B. Scholkopf, and K.R. Mullers, "Fisher discriminant analysis with kernels," in *Proceedings of the 1999 IEEE Signal Processing Society Workshop Neural Networks for Signal Processing IX* ., 1999, pp. 41 –48.

[19] M. Powell, "An efficient method for finding the minimum of a function of several variables without calculating derivatives," vol. 7, pp. 155–162, 1964.

[20] M. Brown, G. Hua, and S. Winder, "Discriminative learning of local image descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 1, pp. 43 –57, 2011.

[21] J. Philbin, M. Isard, J. Sivic, and A. Zisserman, "Descriptor learning for efficient retrieval," in *ECCV*, 2010, pp. 677–691.

[22] B. Kulis and K. Grauman, "Kernelized locality-sensitive hashing for scalable image search," in *ICCV*, 2009, pp. 2130 –2137.

[23] Y. Weiss, A.B. Torralba, and R. Fergus, "Spectral hashing," in *NIPS*, 2008, pp. 1753–1760.

[24] J. Yagnik, D. Strelow, D.A. Ross, and Ruei sung Lin, "The power of comparative reasoning," in *ICCV*, 2011, pp. 2431 –2438.

[25] A.Z. Broder, "On the resemblance and containment of documents," in *Compression and Complexity of Sequences 1997. Proceedings*, 1997, pp. 21 –29.

[26] A.Z. Broder, M. Charikar, A. Frieze, and M. Mitzenmacher, "Min-wise independent permutations," *Journal of Computer and System Sciences*, vol. 60, pp. 327–336, 1998.

[27] Martin A. Fischler and Robert C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[28] Y. Jia, F. Nie, and C. Zhang, "Trace ratio problem revisited," *IEEE Transactions on Neural Networks*, vol. 20, no. 4, pp. 729–735, 2009.

[29] Y. Yang, F. Nie, D. Xu, J. Luo, Y. Zhuang, and Y. Pan, "A multimedia retrieval framework based on semi-supervised ranking and relevance feedback," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 4, pp. 723–742, 2012.

[30] C. Strecha, A. Bronstein, M. Bronstein, and P. Fua, "LDAHash: Improved matching with smaller descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, pp. 66–78, 2012.

[31] G. Lu, N. Sebe, C. Kambhamettu, and C. Xu, "Memory efficient large-scale image-based localization," *Journal of Multimedia Tools and Applications*, 2014.

[32] S. Arya, D.M. Mount, N.S. Netanyahu, R. Silverman, and A.Y. Wu, "An optimal algorithm for approximate nearest neighbor searching fixed dimensions," *Journal of the ACM (JACM)*, vol. 45, no. 6, pp. 891–923, 1998.

[33] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon, "Bundle adjustmenta modern synthesis," in *Vision algorithms: theory and practice*, pp. 298–372. 2000.