Constrained Convolutional Sparse Coding for Parametric Based Reconstruction Of Line Drawings

Sara Shaheen, Lama Affara, and Bernard Ghanem
King Abdullah University of Science and Technology (KAUST), Saudi Arabia
sara.shaheen@kaust.edu.sa, lama.affara@kaust.edu.sa, beranrd.ghanem@kaust.edu.sa

Abstract

Convolutional sparse coding (CSC) plays an essential role in many computer vision applications ranging from image compression to deep learning. In this work, we spot the light on a new application where CSC can effectively serve, namely line drawing analysis. The process of drawing a line drawing can be approximated as the sparse spatial localization of a number of typical basic strokes, which in turn can be cast as a non-standard CSC model that considers the line drawing formation process from parametric curves. These curves are learned to optimize the fit between the model and a specific set of line drawings. Parametric representation of sketches is vital in enabling automatic sketch analysis, synthesis and manipulation. A couple of sketch manipulation examples are demonstrated in this work. Consequently, our novel method is expected to provide a reliable and automatic method for parametric sketch description. Through experiments, we empirically validate the convergence of our method to a feasible solution.

1. Introduction

Sketch vectorization, which is defined as representing a sketch as a number of connected parameterized curves, is of great demand in the graphics community. It enables sketch transformations that are invariant to scale and non-rigid body transformations. It also fuels various applications related to sketch style manipulation, analysis and synthesis [21]. Previous work on those topics tends to avoid the non-trivial geometric solutions of sketch vectorization [20] and replaces it with some manual interaction to access strokes of sketches (strokes are the building blocks of a sketch). This involves pre-processing steps such as building large libraries of strokes or digitally collecting sketches using the Wacome device [10, 19]. As such, utilizing existing modern computing algorithms for *automatic* sketch parametric representation is a valuable contribution to the graphics research community.

In this work, we propose an extension to the formulation of the well-known convolutional sparse coding (CSC) model, such that it can better model the reconstruction process of line drawings. This can be done by constraining the filters in CSC to a pre-defined group of parametric curves. This results in learned filters that, in their content, describe the sketch geometrically as a set of curves. Such representation can be utilized for sketch vectorization. We believe that our work is the first to introduce geometry constraints to the standard CSC formulation.

CSC represents an image by a sum of convolutional responses generated from image filters and sparse maps. These filters constitute a CSC dictionary $\mathbf{D} = \{\mathbf{d}_1, .., \mathbf{d}_K\}$, and when convolved with their corresponding sparse maps $\mathbf{Z} = \{\mathbf{z}_1, .., \mathbf{z}_K\}$, the image is reconstructed. Given an input image $\mathbf{x}$, the CSC optimization is described in Eq (1).

$$\begin{aligned} \underset{\mathbf{d}_k, \mathbf{z}_k\ \forall k}{\arg\min}\ & \frac{1}{2}||\mathbf{x} - \sum_{k=1}^{K} \mathbf{d}_k * \mathbf{z}_k||_2^2 + \beta \sum_{k=1}^{K} ||\mathbf{z}_k||_1 \\ \text{s.t.}\ & ||\mathbf{d}_k||_2^2 \leq 1\ \ \forall k \in \{1, ..., K\} \end{aligned} \tag{1}$$

CSC approximates how sketches are drawn. If the filters in the dictionary are generated from actual parametric strokes, then the summation of convolution responses between these filters and the sparse maps specify *where* in the image a stroke emerges and the set of filters reflect *what* the response (stroke) will look like. We believe that this model approximates how an artist would draw a sketch; however, a more accurate depiction of this process would be to replace the summation with a spatially localized max operation (representing a union instead of a sum). Unfortunately, this higher fidelity model makes the optimization significantly harder to do, although it could still be solved using standard fixed point (alternating) optimization techniques popularly used for bi-convex problems.

To validate our choice of CSC for the task of line drawing analysis, we conducted a user experiment. Using a Wacom and a stylus pen, three artists are asked to freely draw a given sketch. We parse the Wacom data files containing the sketches and divide the sketches into square areas of

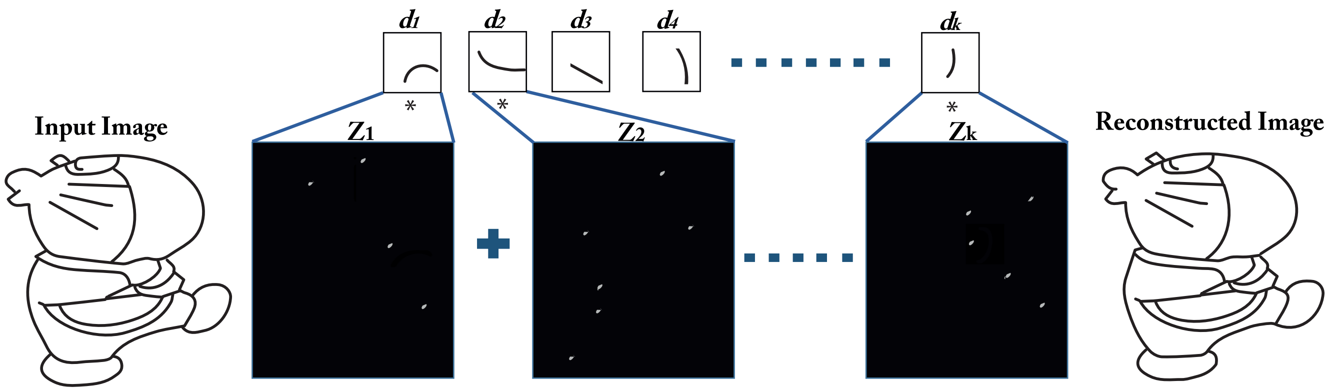

Figure 1: Constrained Convolutional Sparse Coding (CSC) Model. An input image is represented by a sum of dictionary elements belonging to set of parametric curves convolved with corresponding sparse maps and resulting in a reconstructed image that is similar to the input image.

30×30 pixels, where the whole image is of 300×300 pixel resolution. For each square region, we count the average number of times an artist passes his/her pen through this area. This is done using the spatial and temporal information generated from the Wacom device. We eliminate areas that contain no strokes. According to this experiment, the sparsity of strokes used in patches of a sketch reaches a reasonable amount with only 1-pass with the stylus pen over a squared region 60% of the times. This observation validates the use of sparse response maps, thus, justifying the overall use of CSC.

Contributions. In this paper, **(1)** we propose a new method for automatic parametric sketch representation inspired by the well known CSC model. **(2)** We reformulate the original CSC function by adding a constraint on the learned filters to belong to a defined set of parametric curves. Figure 1 gives an illustrative example of how our proposed constrained CSC model is used for line drawing reconstruction using filters that are learnt from a set of parametric curves. We solve the resulting non-convex optimization using ADMM. While proof of convergence is left for future work, our experiments show that our ADMM solver converges to a good feasible solution.

2. Related Work

In this section, we highlight different applications and previous computational advances of the CSC model along with related work on sketch vectorization, synthesis and manipulation techniques, which our work is related to.

CSC Applications and Advancements. Recently, research on CSC has taken two main directions. The first direction focuses on applying CSC to a wide range of computer vision problems such as image processing [8, 2, 7, 24, 11], designing new deep learning architectures [17], computational imaging [13], tracking [25] and structure for motion [27]. The other CSC research direction focuses on finding an efficient solution to the CSC problem. This direction is driven by the high computational demands of minimizing the non-convex CSC objective. Moreover, CSC sparse dictionary learning algorithms are special as they enable for diverse translation-invariant image patches through using the convolution operator in its image representation. Consequently, the literature witnessed seminal advances in CSC as found in [5, 16, 12]. Due to the diagonalization property of circulant matrices in the Fourier domain, speeding up the CSC solution has seen much progress. In fact, Bristow *et al.* [5] propose to model the optimization as two convex subproblems that are solved iteratively in a fixed point strategy. Each subproblem is solved in the Fourier domain using the Alternating Direction Method of Multipliers (ADMM) [26]. Following that, Bristow *et al.* [6] provide a thorough discussion on a number of optimization methods for solving convolution problems and their applications. To further improve efficiency, Kong *et al.* [16] describe an optimization approach that exploits the separability of convolutions across bands in the Fourier domain in order to make dictionary learning more efficient.

Despite all of the mentioned advancements, the problem is still computationally heavy due to the high cost of solving large linear systems. Motivated by that, Heide *et al.* [12] propose a new objective function which transforms the original constrained problem into an unconstrained problem by encoding the constraints in the objective using some indicator functions. Their proposed objective is further split to a set of convex functions that are easier to optimize separately. Moreover, they propose a diagonal mask matrix to the objective to handle boundary artifacts that materialize in the Fourier domain. In addition, Vorsel *et al.* [22] propose a non-iterative method in the Fourier domain for computing the inversion of the convolutional operator using the matrix

inversion lemma. Finally, a recent work of Bibi *et al*. [4] propose a new formulation of CSC that can handle an arbitrary order tensor of data. This, in return, is important to learning multidimensional dictionaries and sparse codes for the reconstruction of multi-dimensional data.
In this work, we propose a novel constrained patch learning CSC model, which learns parametric filters for line drawings reconstruction. To the best of our knowledge, we are the first to introduce a constrained parametric model for CSC. Inspired by [12], we solve the constrained problem using the fixed point strategy, which leads to two subproblems that can be solved in the Fourier domain. Each subproblem is approached using ADMM. As mentioned earlier, our constrained model is expected to facilitate sketch vectorization, synthesis and manipulation which are vital for sketch based computer graphics applications.

Sketch Synthesis and Stroke Manipulation. To start any sketch manipulation process, a method to describe a sketch in terms of its stroke segments is the first stage in the pipeline. An example of an automatic sketch vectorization method is by Noris *et al*. [20]. They propose an automated method to vectorize clean line drawings as connected Bézier curve paths. Their approach is based on extracting strokes centerlines as indicated from the image gradient along a stroke segment. This problem is not trivial to solve geometrically because even when input drawings are comprised of clean and high-contrast lines, inherent ambiguities make vectorization difficult. Consequently, researchers tend to rely on off-the-shelf commercial products, as what is found in Adobe illustrator [1]. Other approaches are based on collecting sketches digitally using the Wacom device, which makes accessing per stroke information and expressing it geometrically a straight forward process [19, 3]. Our proposed work can be viewed as an addition to this rich literature. It provides a fully automated model to represent a sketch geometrically and to provide an automatic mapping to where strokes should be placed in the sketch utilizing the unique formulation of CSC. This will enable more efficient and reliable sketch style analysis, synthesis and manipulation.

Automatic stroke manipulation and regeneration has been researched for a few decades. Thus, it is impossible to cover every related work. For a more detailed representation of the literature landscape, we refer the reader to two survey papers [18, 14]. They are concerned with artistic sketch regeneration and stylization techniques using 2D input images or videos of non-photo realistic rendering (NPR).

3. Constrained Parametric CSC Optimization

In this section, we give a brief overview of the standard CSC model. Then, we reformulate the problem to include our proposed parametric projection constraint. In our work, we consider the CSC problem with circular boundary conditions as discussed in the work of Brisow *et al*. [5].

3.1. Standard CSC Model

The CSC problem is generally expressed in Eq (1), where $\mathbf{d}_k \in \mathbb{R}^M$ are the vectorized 2D patches representing K dictionary elements, and $\mathbf{z}_k \in \mathbb{R}^D$ are the vectorized sparse maps corresponding to each of the dictionary elements. The data term represents the image $\mathbf{x} \in \mathbb{R}^D$ as the sum of 2D convolutions of the dictionary elements with the sparse maps, and the ℓ_1 term is added to enforce sparsity onto the feature maps with β controlling the level of sparsity. The CSC objective function can be easily extended to multiple training images, where K sparse maps are learned for each image and all images share the same K dictionary elements. The objective in Eq (1) is not convex, yet it is bi-convex. So, solving it for one variable while keeping the other fixed leads to two convex subproblems: the coding subproblem and the dictionary learning subproblem.

Learning Subproblem. The learning subproblem is shown in Eq (2). It learns the dictionary elements for a set of sparse feature maps.

$$\arg\min_{\mathbf{d}} \ \frac{1}{2}\|\mathbf{x}-\mathbf{Z}\mathbf{d}\|_2^2 \quad \text{s.t.} \ \ \|\mathbf{d}_k\|_2^2 \leq 1 \ \ \forall k \tag{2}$$

Here, $\mathbf{Z} = [\mathbf{Z}_1 \dots \mathbf{Z}_K]$ is a concatenation of Toeplitz convolution matrices of the sparse maps, and $\mathbf{d} = [\mathbf{d}_1^T \dots \mathbf{d}_K^T]^T$ is a concatenation of all the dictionary elements.
Coding Subproblem. The coding subproblem is detailed in Eq (3). It computes optimal sparse maps for a set of dictionary elements.

$$\arg\min_{\mathbf{z}} \ \frac{1}{2}\|\mathbf{x}-\mathbf{D}\mathbf{z}\|_2^2 + \beta\|\mathbf{z}\|_1 \tag{3}$$

Similar to above, $\mathbf{D} = [\mathbf{D}_1 \dots \mathbf{D}_K]$ is a concatenation of Toeplitz convolution matrices of the dictionary elements, and $\mathbf{z} = [\mathbf{z}_1^T \dots \mathbf{z}_K^T]^T$ is a concatenation of the vectorized sparse maps.
One approach to solve each of the above subproblems is to cast it in an ADMM framework. Also, it can be solved efficiently in the Fourier domain. For more details about the CSC model and for a discussion on various methods to solve it, we refer the reader to the work of Bristow *et al*. [6].

3.2. Constrained Parametric CSC

In this section, we describe the variations we make on the standard CSC model to solve for a constrained parametric CSC (*i.e*. to allow for filters to be constrained to a set of parametric curves). This reformulation enforces geometrical properties on the filters during the training step. As we

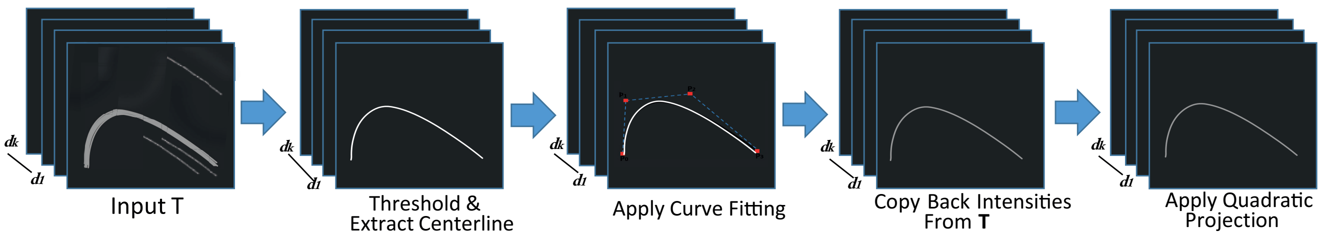

Figure 2: The work flow of parametric curve projection for each filter patch: (1) given an input filter patch T ;(2) extract the largest set of connected pixels with intensity values above a threshold and then extract its centerline; (3) apply cubic Bézier curve fitting; (4) copy the intensity values of pixels where the fitted curve located at T to the generated curve patch; (5) apply quadratic projection to the curve patch.

explain the model, it will become obvious how such reformulation enables a parametric curve representation of the reconstructed line drawings. We reformulate the CSC problem by adding a parametric constraint as follows:

$$\begin{aligned} \underset{\mathbf{d}_k,\mathbf{z}_k\ \forall k}{\arg\min}\ & \frac{1}{2}\|\mathbf{x}-\sum_{k=1}^{K}\mathbf{d}_k * \mathbf{z}_k\|_2^2 + \beta\sum_{k=1}^{K}\|\mathbf{z}_k\|_1 \\ \text{s.t.}\ & \mathbf{d_k}\in S_b;\ \forall k\in\{1,...,K\} \end{aligned} \tag{4}$$

This constraint is added to the learning subproblem so that the learned dictionary patches are geometrically constrained to belong to a set of parametric curves S_b. We define the set of parametric curves as follows:

$$S_b = \left\{\mathbf{a} : \|\mathbf{a}\|_2^2 \le 1 \quad s.t.\ \text{support of } \mathbf{a} \in f(t)\right\}, \tag{5}$$

where the support of $\mathbf{a}$ denotes the set of pixel indices with non-zero intensity in $\mathbf{a}$ and $f(t)$ is the cubic Bézier curve function in Eq (6).

$$f(t) = \sum_{i=0}^{3}\mathbf{p}_i B_i^3(t), \quad t\in[0,1], \tag{6}$$

where $\mathbf{p}_i$ is the i^{th} control point of the 2D curve, and $B_i^3(t)$ is the cubic Bernstein polynomial (i.e. $(1-t)^3,\ 3t(1-t)^2, 3t^2(1-t),\ t^3$) [23]. By manipulating the control points of a Bézier curve, it can be intuitively deformed. Cubic Bézier curves are enough to model various curve shapes and only four control points $\{\mathbf{p}_i\}_{i=0}^3$ are needed as shown in Figure 3. Unlike normal functions, parametric Bézier curves do not define a y coordinate in terms of an x coordinate. Instead, they link the values to *control* variables. If we vary the value of t, then with every change we get two new values, which we can use as (x, y) coordinates in the curve plot. The variable t can be viewed as the variable which we sample the curve along as its value changes over a defined range [9]. We choose cubic Bézier curves to represent underlying strokes in line drawings, since they are widely used in computer graphics and geometric modeling applications that focus on smooth curves [23, 9].

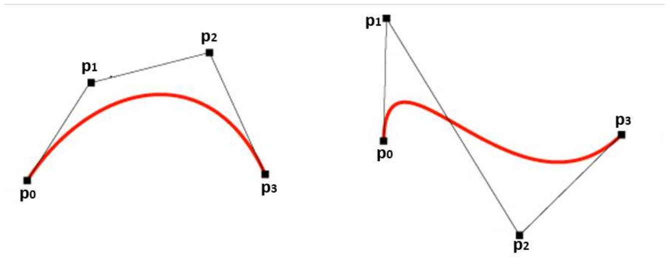

Figure 3: Examples of cubic Bézier curves. Four control points are enough to represent the curve.

Learning Subproblem. Changes on the standard CSC are only reflected in the dictionary learning subproblem. The learning subproblem corresponding the constrained parametric CSC is shown in Eq (7). The inequality constraint on the dictionary elements shown in Eq (1) is now embedded as part of the parametric projection proximal of the set S_b as defined earlier in Eq (5). To be able to solve this subproblem using ADMM, we add the second equality constraint and introduce an auxiliary variable $\mathbf{u}$ as follows:

$$\underset{\mathbf{d},\mathbf{u}}{\arg\min}\ \frac{1}{2}\|\mathbf{x}-\mathbf{Zd}\|_2^2\ \text{ s.t.} \begin{cases}\mathbf{u_k}\in S_b \\ \mathbf{u_k}=\mathbf{d_k}\end{cases} \quad \forall k \tag{7}$$

Using an indicator $\mathbb{I}$ for the constraint, we can move it into the objective yielding the following objective with one constraint, which constitutes our proposed constrained parametric CSC:

$$\begin{aligned} \underset{\mathbf{d},\mathbf{u}}{\arg\min}\ & \frac{1}{2}\|\mathbf{x}-\mathbf{Zd}\|_2^2 + \sum_{k=1}^{K}\mathbb{I}_{\{\mathbf{u_k}\in S_b\}} \\ \text{s.t.}\ & \mathbf{d_k}=\mathbf{u_k};\ \forall k\in\{1,...,K\} \end{aligned} \tag{8}$$

To apply ADMM to Eq (8), we define the augmented Lagragian function $\mathcal{L}$ as follows:

$$\mathcal{L}(\mathbf{d},\mathbf{u},\boldsymbol{\lambda}) := \frac{1}{2}||\mathbf{x}-\mathbf{Z}\mathbf{d}||_2^2 + \sum_{k=1}^{K}\mathbb{I}_{\{\mathbf{u_k}\in S_b\}} + \boldsymbol{\lambda}^{\boldsymbol{T}}(\mathbf{u}-\mathbf{d}) + \frac{\rho}{2}||\mathbf{u}-\mathbf{d}||_2^2 \tag{9}$$

where $\boldsymbol{\lambda}$ is the Lagrangian dual variable of the constraint. The iterative ADMM steps to minimize Eq (9) w.r.t. the primal variables $(\mathbf{d},\mathbf{u})$ are described in Algorithm1. ADMM updates are performed by optimizing for the variables $\mathbf{d}$ and $\mathbf{u}$ one at a time, while keeping the other. Then, the dual variable $\boldsymbol{\lambda}$ is updated using gradient ascent. Now, we list the optimization solutions to update the primal variables in the ADMM algorithm.

Algorithm 1 ADMM for learning subproblem

1: Set ADMM optimization parameter $\rho > 0$
2: Initialize variables $\mathbf{d}$, $\mathbf{u}$, $\boldsymbol{\lambda}$
3: **while** not converged **do**
4: $\mathbf{d^{t+1}} = \arg\min_d \frac{1}{2}||\mathbf{x}-\mathbf{Z}\mathbf{d}||_2^2 - \boldsymbol{\lambda}^{\boldsymbol{T}}\mathbf{d}) + \frac{\rho}{2}||\mathbf{u}-\mathbf{d}||_2^2$
5: $\mathbf{u^{t+1}} = prox_{\frac{f}{\rho}}\left(\mathbf{d^{t+1}} + \frac{\boldsymbol{\lambda^t}}{\rho}\right)$
where $f(u) = \sum_{k=1}^{K}\mathbb{I}_{\{\mathbf{u_k}\in S_b\}}$
6:
7: $\boldsymbol{\lambda^{t+1}} = \boldsymbol{\lambda^t} + \rho\left(\mathbf{u^{t+1}} - \mathbf{d^{t+1}}\right)$
8: $\rho = \rho + c$
9: Output solution variables

Update $\mathbf{d^{t+1}}$: At the t^{th} iteration, the update of this variable is a simple least squares problem, which has the following solution: $\mathbf{d^{t+1}} = \left(\mathbf{Z}^T\mathbf{Z} + \rho I\right)^{-1}\left(\mathbf{Z}^T\mathbf{x} + \boldsymbol{\lambda^t} + \mathbf{u^t}\right)$. We initialize $\mathbf{d}$ as randomly generated parametric curves.

Update $\mathbf{u^{t+1}}$: Updating this variable is done through the proximal operator $g(.)$ for the indicator function of set S_b. Mathematically, the underlying optimization is:

$$g(\mathbf{T}) = \arg\min_{\mathbf{u}} \ \frac{\rho}{2}\|\mathbf{u}-\mathbf{T}\|_2^2 + \sum_{k=1}^{K}\mathbb{I}_{\{\mathbf{u_k}\in S_b\}}$$

where $\mathbf{T} = \mathbf{d^{t+1}} + \frac{\boldsymbol{\lambda^t}}{\rho}$. Consequently, the goal of our parametric projection proximal operator $g(.)$ is to find the best cubic Bézier curve, which defines the support of an image patch *closest* in intensity to $\mathbf{T}$. In this way, we are geometrically guiding the indices of the non-zero elements of the learned filters to form a proper Bézier curve. To achieve that, we develop a method to perform this parametric projection to a Bézier curve, as illustrated in Figure 2. First, we extract the largest set of connected pixels with intensities above the average intensity of $\mathbf{T}$. We then apply morphological skeletonization [15] to extract the centerline of those high intensity pixels. This will generate a skeleton representation of the extracted pixels. Next, we fit a cubic Bézier curve on the extracted skeleton using Eq (6) and then copy the intensity values at these pixels from $\mathbf{T}$. Lastly, we project the resulting image patch onto the unit ball to enforce that $\|\mathbf{u_k}\|_2^2 \leq 1;\ \forall k \in \{1,...,K\}$. This ensures that the solution vector is of the proper scale. It is important to mention that the set S_b is a non-convex set, thus, the parametric projection is not necessarily globally optimal. However, our experiments, as we demonstrate later, show that our multi-stage technique to define the proximal operator $g(.)$ converges to feasible solutions.

Coding Subproblem. The coding subproblem of the constrained parametric CSC is the same as in the standard CSC discussed in Section 3.1. It is solved by applying ADMM where the update steps involve solving a linear system and a proximal operator for the ℓ_1 norm. The linear system is solved efficiently in the Fourier doamin making use of the circulant structure of the convolution matrices. The proximal operator, on the other hand, is solved using soft-thresholding. We refer the reader to the work of Heide *et al.* [12] for details on how to solve the coding subproblem and for overall complexity analysis.

4. Results

In this section, we provide an assessment and a discussion of the proposed constrained parametric CSC. We start with an overview of the implementation details and the parameters selected throughout our experiments. Then, we discuss the convergence of our technique and provide visual evaluation of the learned parametric filters, as compared to standard CSC, along with image reconstruction quality. Towards the end of this section, we present some qualitative examples of sketch synthesis and manipulation using our parametric model.

4.1. Implementation Details

The line drawings used in our experiments are from the free style line drawings dataset of Shaheen *et al.* [21]. It comprises 70 clean line drawings of multiple experienced artists. Before learning the dictionary elements, we apply contrast normalization to the sketch images to generate normalized gray-scale images with black background and white sketch drawings in the foreground. All these images of 300×300 spatial resolution. It is important to mention that the quality of sketches used to learn the dictionary elements is crucial. The technique performs best when using high resolution images of line drawings to learn the dictionary.

The number of dictionary elements is chosen to be $K = 100$, where each element has a spatial size of 31×31 pixels. We found that this size of individual dictionary elements is representative enough of the different curves in a line drawing, thus, allows for a various collection of Bézier curves to be fit to the learned image filters. Smaller dictionary elements, on the other hand, lead to non-significant details per dictionary patch while larger filter sizes carry lots of details such that it is impossible to fit a single Bézier curve per dictionary patch. We initialize each dictionary element as a normalized image with black background and a white foreground with a support that traces a *randomly generated* Bézier curve.

Similar to unconstrained CSC, the sparsity coefficient β controls the level of sparsity in the coding sub-problem. High sparsity (i.e. large β) ensures that only a sparse number of strokes (dictionary elements) are placed in the reconstructed sketch. In training, this leads to a better geometrical representation of the sketch; however, the sketch reconstruction quality suffers. On the other hand, low sparsity (i.e. small β) leads to less geometrically representative strokes but better reconstruction quality. Through cross-validation, we find $\beta = 0.6$ strikes a reasonable trade-off between representative parametric curves and reconstruction quality. Moreover, for the ADMM learning steps we start with a value of $\rho = 0.1$ which increases by 2% at every iteration.

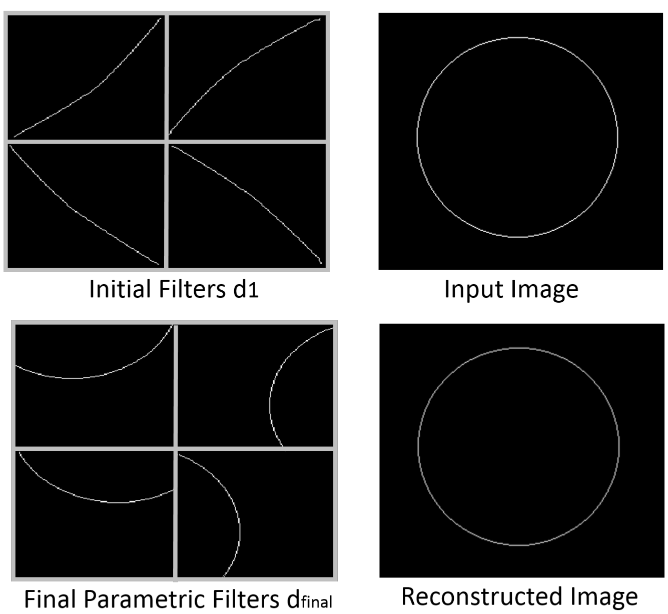

Figure 4: An example of reconstructing a circle using our constrained parametric CSC. Filters in the last iteration have deformed to a new set of curves that are clearly representing the reconstructed image and describe it geometrically. Furthermore, the reconstructed image quality is preserved when compared with the input image.

4.2. Simple Reconstruction Example

In Figure 4, we present an example for constructing a simple primitive shape (a circle) using $K = 4$ dictionary elements using our constrained parametric CSC model. Starting with initial dictionary patches that are almost linear Bézier curves, it is obvious how the dictionary patches in the last iteration have changed to represent a new set of curves that are of better representation power for the end shape (a circle). Using this simple example, we demonstrate how our constrained parametric CSC model works in general. The learned parametric filters can now be used to describe the reconstructed image geometrically.

4.3. Evaluation of Constrained Parametric CSC

Multiple intuitive questions arise while discussing our proposed parametric CSC, which we address in this section. First, we study the reconstruction quality of the parametric CSC and show how the dictionary elements look like upon convergence. As illustrated in the first 2 columns of Figure 5, learned dictionary elements (strokes) change significantly when compared against the initial random curves. Such change yields to dictionary elements that represent the image geometrically through the parametrically defined filters. Also, we see that the reconstruction quality is good with a Peak Signal-to-Noise Ratio (PSNR) of 29dB.

We also compare our parametric model with standard CSC. Dictionary elements learned on the standard model are illustrated in Figure 5. It is obvious that they do not carry clear geometric information evident in line drawings and they cannot be used for parametric representation of sketches. However, the reconstruction quality of the standard CSC is higher than our parametric CSC with PSNR=40dB. This is expected, since standard CSC ideally offers a lower bound objective to that of our constrained parametric CSC. It is important to mention that all experiments are conducted at the same sparsity level (of 60%) for the sparse codes.

To evaluate the need for applying the proximal operator $g(.)$ within the ADMM framework, we apply this operator to fit cubic Bézier curves to the filters learned by standard CSC. As shown in Figure 5, these learned filters are not representative of the content of the line drawing, since it is hard to fit one curve to each filter generated through standard CSC. Projecting these learned filters onto S_b generates subpar results that are not reflective of the strokes drawn by the artist. In addition, the reconstruction quality using these projected filters is 18dB lower than our constrained parametric CSC. Therefore, we conclude that our method is capable of generating high reconstruction quality line drawings, while constraining its filters to be parametric Bézier curves.

4.4. Convergence

Our constrained parametric CSC optimization is non-convex in general. This is because the set S_b in Eq (7) is

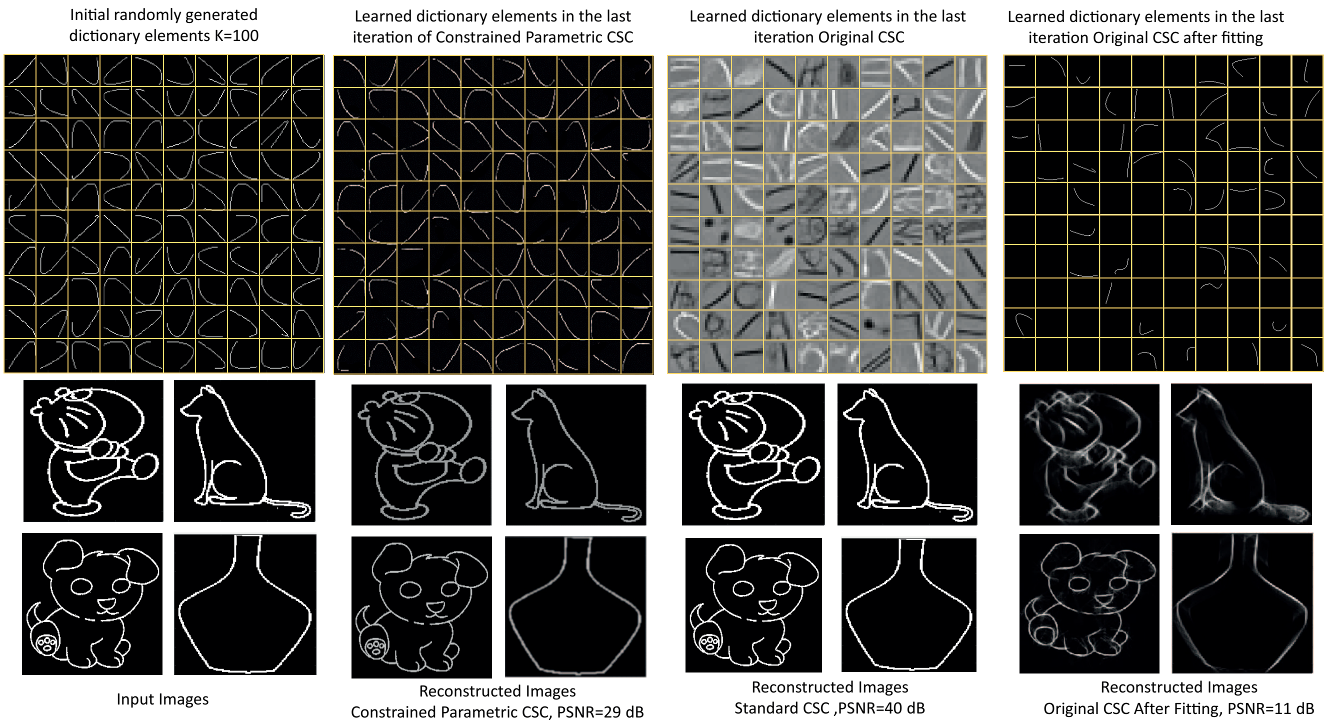

Figure 5: A comparison of the the dictionary learning elements and the quality of the reconstructed images between (1) the constrained parametric CSC, (2) the standard CSC and (3) the standard CSC with curves fitting in the last iteration.

non-convex, *i.e*. the convex combination of any two patches with cubic Bézier curve support does necessarily generate a patch whose support is a cubic Bézier curve. Despite this non-convexity, our experiments show that the ADMM iterations do empirically converge (refer to Figure 6). The plots in Figure 6 are generated when constructing multiple line drawings with 300×300 pixel resolution and with $K = 100$ dictionary elements of resolution 31×31 pixels, similar to Figure 5. The first plot on top shows how the overall objective value of parametric CSC decreases as the fixed point (or alternative optimization) iterations progress, refer to Eq (4). The second plot is an evidence of convergence for a single learning subproblem that uses ADMM, Eq (8). The learning objective eventually decreases and then saturates after a certain number of iterations. The early increase in this plot is due to the fact that the current solution is *not* feasible, *i.e*. the variables $\mathbf{d}$ and $\mathbf{u}$ are far away from each other. This is clear from the bottom plot, which measures how the primary variables $\mathbf{d}$ and $\mathbf{u}$ are getting closer as the ADMM iterations progress for the learning subproblem. As this distance measure reaches zero, the resulting variables are guaranteed to be the same, thus, reaching a feasible solution to the original constrained problem in Eq (7).

As in any non-convex optimization problem, initialization of the optimization variables is an important element to solve the problem so it converges to a feasible solution. Our parametric optimization, on the other hand, is not significantly sensitive to initialization. This is evident on how our optimization is able to converge and generate promising results when using randomly generated cubic Bézier curves images as its initial dictionary elements. However, the closer the initialized parametric filters to the actual stroke of the drawings, the faster the convergence and the higher the overall reconstruction quality. Lastly, we leave the proof of convergence of ADMM on this non-convex problem to future work.

4.5. Qualitative Sketch Manipulation Results

As mentioned earlier in Section 1, parametric representation of sketches is vital to enable for automatic sketch synthesis and manipulation. Here we demonstrate a couple of line drawing stylization examples (i.e. embedding a new style into a line drawing) on one drawing as shown in Figure 7. Given the parametric filters learned by our constrained parametric CSC model, as in Figure 5, we can now directly apply various geometric transformations to those parametric curves. This will generate a set of new dictionary elements which we use to reconstruct the same line drawing using the same sparse feature maps inferred from the constrained parametric CSC model. Consequently, this will embed a new style to the strokes of the line drawing.

The first stylization example we include in this work aims to increase the thickness of strokes in the sketched line drawing. This is simply achieved by increasing the thickness of

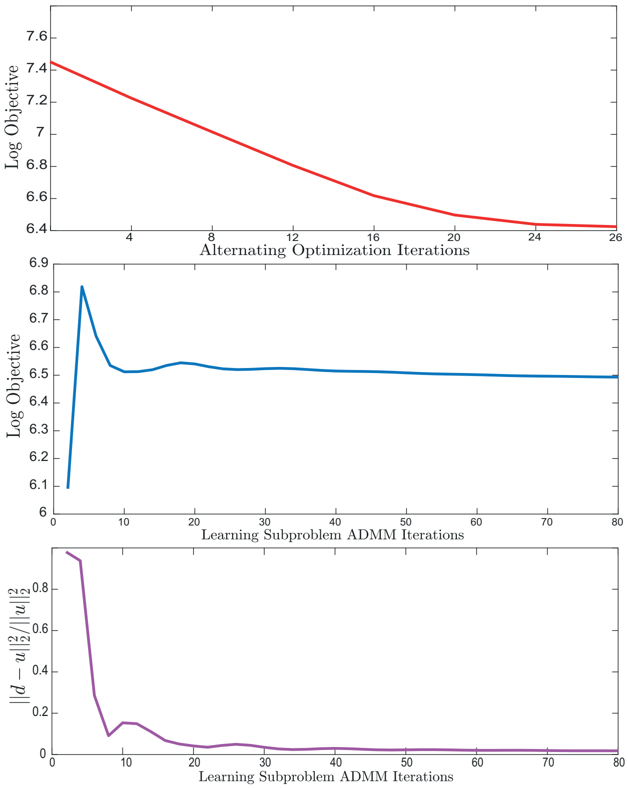

Figure 6: Convergence experiments. The first plot shows how the objective function value decreases over alternating optimization iterations. The second plot shows how the ADMM subproblem objective value decreases over the dictionary learning iterations and then saturated. The third plot shows how the primary variables $\mathbf{u}$ and $\mathbf{d}$ are getting closer to each other until the difference between them is almost 0.

the parametric curve in each patch by generating the same curve and take two copies of it and place them one pixel above and one pixel below the original curve. The process of shifting and copying the curve can be repeated as many times as needed around the original parametric curve until getting the desired thickness. Then we replace the parametric learned filters with the modified ones and reconstruct the line drawings using the earlier inferred feature maps to embed this new style. Thickness stylization result is shown on the image labeled as **Stylized Image 1** of Figure 7. The second example is generated by replacing each parametric curve with a sine wave over the curve path. This can simply be accomplished by replacing each parametric curve $f(t)$ in Eq (6) with $f(t)+\sin(t)$. Similarly, this will generate a new set of dictionary elements which are used to reconstruct the line drawing to embed the new style. This example is shown on the image labeled as **Stylized Image 2** of Figure 7.

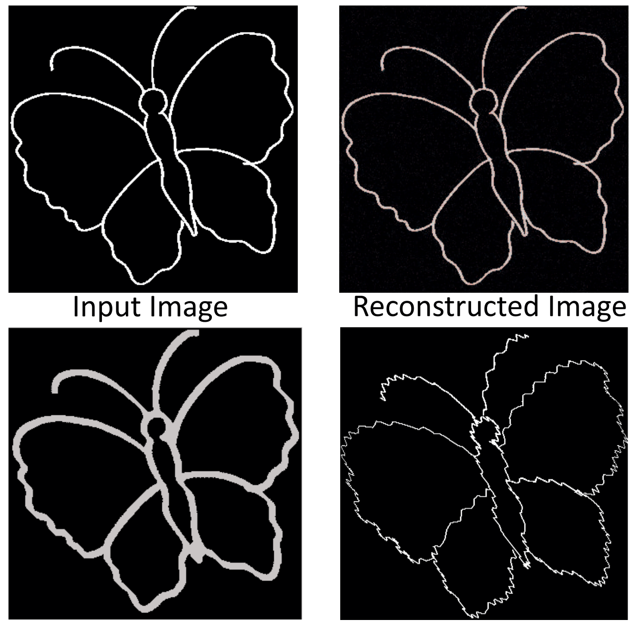

Figure 7: Examples demonstrating how sketch stylization and manipulation is enabled upon having parametric representation of sketches.

5. Conclusion and Future Work

In this work, we present an automatic method for representing a sketch geometrically as a set of parametric curves. It is based on utilizing the well known convolutional sparse coding (CSC) model. This is based on our observation that CSC is closely related to the line drawing process. The i^{th} map $\mathbf{z}_i$ can be viewed as a sparse score map, which scores each pixel based on whether the filter or stroke $\mathbf{d}_i$ is localized there or not. This scenario resembles how one would draw a sketch, *i.e.* making the decision what stroke to draw and where to place it. To fit for the task of line drawing, we reformulate the standard CSC model such that the dictionary elements are constrained to belong to a set of images, whose support is a parametric cubic Bézier curve. As such, sketches are represented geometrically through the learned CSC filters. Such reformulation leads to a non-convex learning subproblem of the CSC, which we solve using ADMM. Although convergence is not theoretically guaranteed in this non-convex case, experiments show that our algorithm converges to a good solution.

For future work, we aim to utilize the learned parametric dictionary elements to help design fully automated techniques for sketch synthesis and style transfer.

Acknowledgments. This work was supported by competitive research funding from King Abdullah University of Science and Technology (KAUST).

References

[1] ADOBE. Illustrator. 2010. 3

[2] M. Aharon, M. Elad, and A. M. Bruckstein. On the uniqueness of overcomplete dictionaries, and a practical way to retrieve them. *Linear algebra and its applications*, 416(1):48–67, 2006. 2

[3] I. Berger, A. Shamir, M. Mahler, E. Carter, and J. Hodgins. Style and abstraction in portrait sketching. *TOG*, 32:55:1–55:12, 2013. 3

[4] A. Bibi and B. Ghanem. High order tensor formulation for convolutional sparse coding. *ICCV*, 2017. 3

[5] H. Bristow, A. Eriksson, and S. Lucey. Fast Convolutional Sparse Coding. *CVPR*, 2013. 2, 3

[6] H. Bristow and S. Lucey. Optimization Methods for Convolutional Sparse Coding. *arXiv Prepr. arXiv1406.2407v1*, 2014. 2, 3

[7] F. Couzinie-Devy, J. Mairal, F. Bach, and J. Ponce. Dictionary learning for deblurring and digital zoom. *arXiv preprint arXiv:1110.0957*, 2011. 2

[8] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image processing*, 15(12):3736–3745, 2006. 2

[9] G. Farin. From conics to nurbs: A tutorial and survey. *IEEE Comput. Graph. Appl.*, 12(5):78–86, 1992. 4

[10] W. T. Freeman, J. B. Tenenbaum, and E. C. Pasztor. Learning style translation for the lines of a drawing. *ACM Trans. Graph.*, 22(1):33–46, Jan. 2003. 1

[11] S. Gu, W. Zuo, Q. Xie, D. Meng, X. Feng, and L. Zhang. Convolutional sparse coding for image super-resolution. *ICCV*, 2015. 2

[12] F. Heide, W. Heidrich, and G. Wetzstein. Fast and Flexible Convolutional Sparse Coding. *CVPR*, 2015. 2, 3, 5

[13] F. Heide, L. Xiao, A. Kolb, M. B. Hullin, and W. Heidrich. Imaging in scattering media using correlation image sensors and sparse convolutional coding. *Optics express*, 2014. 2

[14] M. Indu and K. Kavitha. Survey on sketch based image retrieval methods. In *ICCPCT*, pages 1–4. IEEE, 2016. 3

[15] B. K. Jang and R. T. Chin. Analysis of thinning algorithms using mathematical morphology. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12, 1990. 5

[16] B. Kong and C. C. Fowlkes. Fast Convolutional Sparse Coding. *Tech. Rep. UCI*, 2014. 2

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 2012. 2

[18] J. E. Kyprianidis, J. Collomosse, T. Wang, and T. Isenberg. State of the art: A taxonomy of artistic stylization techniques for images and video. *TVCG*, 19(5):866–885, May 2013. 3

[19] J. Lu, F. Yu, A. Finkelstein, and S. DiVerdi. Helpinghand: example-based stroke stylization. *TOG*, 31(4):46, 2012. 1, 3

[20] G. Noris, A. Hornung, R. W. Sumner, M. Simmons, and M. Gross. Topology-driven vectorization of clean line drawings. *TOG*, 32, 2013. 1, 3

[21] S. Shaheen, A. Rockwood, and B. Ghanem. Sar: Stroke authorship recognition. *CGF*, 2015. 1, 5

[22] M. Šorel and F. Šroubek. Fast convolutional sparse coding using matrix inversion lemma. *Digital Signal Processing*, 55:44–51, 2016. 2

[23] X. Wei. Rational bezier representation of conics. 1991. 4

[24] J. Yang, J. Wright, T. Huang, and Y. Ma. Image super-resolution as sparse representation of raw image patches. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008. 2

[25] T. Zhang, A. Bibi, and B. Ghanem. In Defense of Sparse Tracking: Circulant Sparse Tracker. *CVPR*, 2016. 2

[26] W. Zhong and J. T.-Y. Kwok. Fast stochastic alternating direction method of multipliers. In *ICML*, pages 46–54, 2014. 2

[27] Y. Zhu and S. Lucey. Convolutional sparse coding for trajectory reconstruction. *PAMI*, 2015. 2