Inverse Compositional Spatial Transformer Networks

Chen-Hsuan Lin   Simon Lucey
The Robotics Institute
Carnegie Mellon University

chenhsul@andrew.cmu.edu   slucey@cs.cmu.edu

Abstract

In this paper, we establish a theoretical connection between the classical Lucas & Kanade (LK) algorithm and the emerging topic of Spatial Transformer Networks (STNs). STNs are of interest to the vision and learning communities due to their natural ability to combine alignment and classification within the same theoretical framework. Inspired by the Inverse Compositional (IC) variant of the LK algorithm, we present Inverse Compositional Spatial Transformer Networks (IC-STNs). We demonstrate that IC-STNs can achieve better performance than conventional STNs with less model capacity; in particular, we show superior performance in pure image alignment tasks as well as joint alignment/classification problems on real-world problems.

1. Introduction

Recent rapid advances in deep learning are allowing for the learning of complex functions through convolutional neural networks (CNNs), which have achieved state-of-the-art performances in a plethora of computer vision tasks [9, 17, 4]. Most networks learn to tolerate spatial variations through: (a) spatial pooling layers and/or (b) data augmentation techniques [16]; however, these approaches come with several drawbacks. Data augmentation (*i.e.* the synthetic generation of new training samples through geometric distortion according to a known noise model) is probably the oldest and best known strategy for increasing spatial tolerance within a visual learning system. This is problematic as it can often require an exponential increase in the number of training samples and thus the capacity of the model to be learned. Spatial pooling operations can partially alleviate this problem as they naturally encode spatial invariance within the network architecture and uses sub-sampling to reduce the capacity of the model. However, they have an intrinsic limited range of tolerance to geometric variation they can provide; furthermore, such pooling operations destroy spatial details within the images that could be crucial to the performance of subsequent tasks.

Instead of designing a network to solely give tolerance to spatial variation, another option is to have the network solve for some of the geometric misalignment in the input images [12, 6]. Such a strategy only makes sense, however, if it has lower capacity and computational cost as well as better performance than traditional spatially invariant CNNs. Spatial Transformer Networks (STNs) [7] are one of the first notable attempts to integrate low capacity and computationally efficient strategies for *resolving* - instead of *tolerating* - misalignment with classical CNNs. Jaderberg *et al*. presented a novel strategy for integrating image warping within a neural network and showed that such operations are (sub-)differentiable, allowing for the application of canonical backpropagation to an image warping framework.

The problem of learning a low-capacity relationship between image appearance and geometric distortion is not new in computer vision. Over three and a half decades ago, Lucas & Kanade (LK) [14] proposed the seminal algorithm for gradient descent image alignment. The LK algorithm can be interpreted as a feed forward network of multiple alignment modules; specifically, each alignment module contains a low-capacity predictor (typically linear) for predicting geometric distortion from relative image appearance, followed by an image resampling/warp operation. The LK algorithm differs fundamentally, however, to STNs in their application: image/object alignment instead of classification.

Putting applications to one side, the LK and STN frameworks share quite similar characteristics however with a criticial exception. In an STN with multiple feed-forward alignment modules, the output image of the previous alignment module is directly fed into the next. As we will demonstate in this paper, this is problematic as it can create unwanted boundary effects as the number of geometric prediction layers increase. The LK algorithm does not suffer from such problems; instead, it feeds the warp parameters through the network (instead of the warped image) such that each subsequent alignment module in the network resamples the original input source image. Furthermore, the Inverse Compositional (IC) variant of the LK algorithm [2] has demonstrated to achieve equivalently ef-

fective alignment by reusing the same geometric predictor in a compositional update form.

Inspired by the IC-LK algorithm, we advocate an improved extension to the STN framework that (a) propagates *warp parameters*, rather than *image intensities*, through the network, and (b) employs the same geometric predictor that could be reapplied for all alignment modules. We propose Inverse Compositional Spatial Transformer Networks (IC-STNs) and show its superior performance over the original STNs across a myriad of tasks, including pure image alignment and joint alignment/classification problems.

We organize the paper as follows: we give a general review of efficient image/object alignment in Sec. 2 and an overview of Spatial Transformer Networks in Sec. 3. We describe our proposed IC-STNs in detail in Sec. 4 and show experimental results for different applications in Sec. 5. Finally, we draw to our conclusion in Sec. 6.

2. Efficient Image & Object Alignment

In this section, we give a review of nominal approaches to efficient and low-capacity image/object alignment.

2.1. The Lucas & Kanade Algorithm

The Lucas & Kanade (LK) algorithm [14] has been a popular approach for tackling dense alignment problems for images and objects. For a given geometric warp function parameterized by the warp parameters $\mathbf{p}$, one can express the LK algorithm as minimizing the sum of squared differences (SSD) objective in the image space,

$$\min_{\Delta\mathbf{p}} \|\mathcal{I}(\mathbf{p} + \Delta\mathbf{p}) - \mathcal{T}(\mathbf{0})\|_2^2, \qquad (1)$$

where $\mathcal{I}$ is the source image, $\mathcal{T}$ is the template image to align against, and $\Delta\mathbf{p}$ is the warp update being estimated. Here, we denote $\mathcal{I}(\mathbf{p})$ as the image $\mathcal{I}$ warped with the parameters $\mathbf{p}$. The LK algorithm assumes a approximate linear relationship between appearance and geometric displacements; specifically, it linearizes (1) by taking the first-order Taylor approximation as

$$\min_{\Delta\mathbf{p}} \left\| \mathcal{I}(\mathbf{p}) + \frac{\partial\mathcal{I}(\mathbf{p})}{\partial\mathbf{p}}\Delta\mathbf{p} - \mathcal{T}(\mathbf{0}) \right\|_2^2. \qquad (2)$$

The warp parameters are thus additively updated through $\mathbf{p} \leftarrow \mathbf{p} + \Delta\mathbf{p}$, which can be regarded as a quasi-Newton update. The term $\frac{\partial\mathcal{I}(\mathbf{p})}{\partial\mathbf{p}}$, known as the steepest descent image, is the composition of image gradients and the predefined warp Jacobian, where the image gradients are typically estimated through finite differences. As the true relationship between appearance and geometry is seldom linear, the warp update $\Delta\mathbf{p}$ must be iteratively estimated and applied until convergence is reached.

A fundamental problem with the canonical LK formulation, which employs addtive updates of the warp parameters, is that $\frac{\partial\mathcal{I}(\mathbf{p})}{\partial\mathbf{p}}$ must be recomputed on the rewarped images for each iteration, greatly impacting computational efficiency. Baker and Matthews [2] devised a computationally efficient variant of the LK algorithm, which they referred to as the Inverse Compositional (IC) algorithm. The IC-LK algorithm reformulates (1) to predict the warp update to the template image instead, written as

$$\min_{\Delta\mathbf{p}} \|\mathcal{I}(\mathbf{p}) - \mathcal{T}(\Delta\mathbf{p})\|_2^2, \qquad (3)$$

and the linearized least-squares objective is thus formed as

$$\min_{\Delta\mathbf{p}} \left\| \mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0}) - \frac{\partial\mathcal{T}(\mathbf{0})}{\partial\mathbf{p}}\Delta\mathbf{p} \right\|_2^2. \qquad (4)$$

The least-squares solution is given by

$$\Delta\mathbf{p} = \left(\frac{\partial\mathcal{T}(\mathbf{0})}{\partial\mathbf{p}}\right)^\dagger (\mathcal{I}(\mathbf{p}) - \mathcal{T}(\mathbf{0})), \qquad (5)$$

where the superscript $\dagger$ denotes the Moore-Penrose pseudo-inverse operator. This is followed by the inverse compositional update $\mathbf{p} \leftarrow \mathbf{p} \circ (\Delta\mathbf{p})^{-1}$, where we abbreviate the notation $\circ$ to be the composition of warp functions parameterized by $\mathbf{p}$, and $(\Delta\mathbf{p})^{-1}$ is the parameters of the inverse warp function parameterized by $\Delta\mathbf{p}$.

The solutions of (2) and (4) are in the form of linear regression, which can be more generically expressed as

$$\Delta\mathbf{p} = \mathbf{R} \cdot \mathcal{I}(\mathbf{p}) + \mathbf{b}, \qquad (6)$$

where $\mathbf{R}$ is a linear regressor establishing the linear relationship between appearance and geometry, and $\mathbf{b}$ is the bias term. Therefore, LK and IC-LK can be interpreted as belonging to the category of cascaded linear regression approaches for image alignment.

It has been shown [2] that the IC form of LK is effectively equivalent to the original form; the advantage of the IC form lies in its efficiency of computing the fixed steepest descent image $\frac{\partial\mathcal{T}(\mathbf{0})}{\partial\mathbf{p}}$ in the least-squares objective. Specifically, it is evaluated on the static template image $\mathcal{T}$ at the identity warp $\mathbf{p} = \mathbf{0}$ and remains constant across iterations, and thus so is the resulting linear regressor $\mathbf{R}$. This gives an important theoretical proof of concept that a fixed predictor of geometric updates can be successfully employed within an iterative image/object alignment strategy, further reducing unnecessary model capacities.

2.2. Learning Alignment from Data

More generally, cascaded regression approaches for alignment can be learned from data given that the distribution of warp displacements is known a priori. A notable

example of this kind of approach is the Supervised Descent Method (SDM) [19], which aims to learn the series of linear geometric predictors $\{\mathbf{R}, \mathbf{b}\}$ from data. The formulation of SDM's learning objective is

$$\min_{\mathbf{R},\mathbf{b}} \sum_{n=1}^{N} \sum_{j=1}^{M} \|\delta\mathbf{p}_{n,j} - \mathbf{R} \cdot \mathcal{I}_n(\mathbf{p}_n \circ \delta\mathbf{p}_{n,j}) - \mathbf{b}\|_2^2, \quad (7)$$

where $\delta\mathbf{p}$ is the geometric displacement drawn from a known generating distribution using Monte Carlo sampling, and M is the number of synthetically created examples for each image. Here, the image appearance $\mathcal{I}$ is often replaced with a predefined feature extraction function (*e.g.* SIFT [13] or HOG [3]) of the image. This least-squares objective is typically solved with added regularization (*e.g.* ridge regression) to ensure good matrix condition.

SDM is learned in a sequential manner, *i.e.* the training data for learning the next linear model is drawn from the same generating distribution and applied through the previously learned regressors. This has been a popular approach for its simplicity and effectiveness across various alignment tasks, leading to a large number of variants [15, 1, 11] of similar frameworks. Like the LK and IC-LK algorithms, SDM is another example of employing multiple low-capacity models to establish the nonlinear relationship between appearance and geometry. We draw the readers' attention to [11] for a more formally established link between LK and SDM.

It is a widely agreed that computer vision problems can be solved much more efficiently if misalignment among data is eliminated. Although SDM learns alignment from data and guarantees optimal solutions after each applied linear model, it is not clear whether such alignment learned in a greedy fashion is optimal for the subsequent tasks at hand, *e.g.* classification. In order to optimize in terms of the *final* objective, it would be more favorable to paramterize the model as a deep neural network and optimize the entire model using backpropagation.

3. Spatial Transformer Networks

In the rapidly emerging field of deep learning among with the explosion of available collected data, deep neural networks have enjoyed huge success in various vision problems. Nevertheless, there had not been a principled way of resolving geometric variations in the given data. The recently proposed Spatial Transformer Networks [7] performs spatial transformations on images or feature maps with a (sub-)differentiable module. It has the effects of reducing geometric variations inside the data and has brought great attention to the deep learning community.

In the feed-forward sense, a Spatial Transformer warps an image conditioned on the input. This can be mathemati-
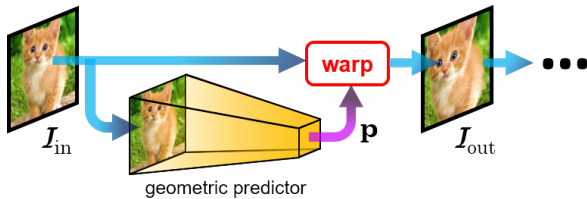

Figure 1: Network module of **Spatial Transformers** [7]. The blue arrows indicate information passing of appearance, and the purple one indicate that of geometry. The yellow 3D trapezoid denotes the geometric predictor, which contains the learnable parameters.

cally written as

$$\mathcal{I}_{\text{out}}(\mathbf{0}) = \mathcal{I}_{\text{in}}(\mathbf{p}), \quad \text{where} \quad \mathbf{p} = f(\mathcal{I}_{\text{in}}(\mathbf{0})). \quad (8)$$

Here, the nonlinear function f is parametrized as a learnable geometric predictor (termed the localization network in the original paper), which predicts the warp parameters from the input image. We note that the "grid generator" and the "sampler" from the original paper can be combined to be a single warp function. We can see that for the special case where the geometric predictor consists of a single linear layer, f would consists of a linear regressor $\mathbf{R}$ as well as a bias term $\mathbf{b}$, resulting the geometric predictor in an equivalent form of (6). This insight elegantly links the STN and LK/SDM frameworks together.

Fig. 1 shows the basic architecture of STNs. STNs are of great interest in that transformation predictions can be learned while also showing that grid sampling functions can be (sub-)differentiable, allowing for backpropagation within an end-to-end learning framework.

Despite the similarities STNs have with classic alignment algorithms, there exist some fundamental drawbacks in comparison to LK/SDM. For one, it attempts to directly predict the optimal geometric transformation with a single geometric predictor and does not take advantage of the employment of multiple lower-capacity models to achieve more efficient alignment before classification. Although it has been demonstrated that multiple Spatial Transformers can be inserted between feature maps, the effectiveness of such employment has on improving performance is not well-understood. In addition, we can observe from (8) that no information of the geometric warp $\mathbf{p}$ is preserved after the output image; this leads to a boundary effect when resampling outside the input source image. A detailed treatment on this part is provided in Sec. 4.1.

In this work, we aim to improve upon STNs by theoretically connecting it to the LK algorithm. We show that employing multiple low-capacity models as in LK/SDM for learning spatial transformation within a deep network yields
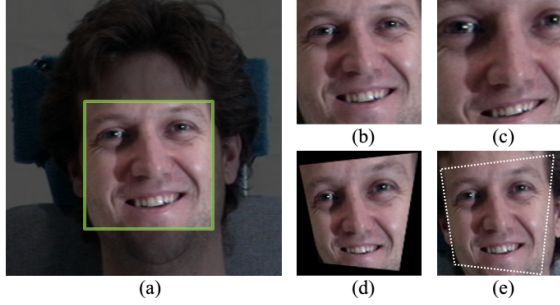
Figure 2: **Boundary effect** of Spatial Transformers on real images. (a) Original image, where the green box indicates the cropped region. (b) Cropped image as the input of the Spatial Transformer. (c) Zoom-in transformation: sampling occurs within the range of the input image. (d)(e) Zoom-out transformation: discarding the information outside the input image introduces a boundary effect (STNs), while it is not the case with geometry preservation (c-STNs). The white dotted box indicates the warp from the original image.

substantial improvement on the subsequent task at hand. We further demonstrate the effectiveness of learning a single geometric predictor for recurrent transformation and propose the Inverse Compositional Spatial Transformer Networks (IC-STNs), which exhibit significant improvements over the original STN on various problems.

4. Inverse Compositional STNs

4.1. Geometry Preservation

One of the major drawbacks of the original Spatial Transformer architecture (Fig. 1) is that the output image samples only from the cropped input image; pixel information outside the cropped region is discarded, introducing a boundary effect. Fig. 2 illustrates the phenomenon.

We can see from Fig. 2(d) that such effect is visible for STNs in zoom-out transformations where pixel information outside the bounding box is required. This is due to the fact that geometric information is not preserved after the spatial transformations. In the scenario of iterative alignment, boundary effects are accumulated for each zoom-out transformations. Although this is less of an issue with images with clean background, this is problematic with real images.

A series of spatial transformations, however, can be composed and described with exact expressions. Fig. 3 illustrates an improved alignment module, which we refer to as compositional STNs (c-STNs). Here, the geometric transformation is also predicted from a geometric predictor, but the *warp parameters* **p** are kept track of, composed, and passed through the network instead of the *warped images*. It is important to note that if one were to incorporate a cascade of multiple Spatial Transformers, the geometric transforma-
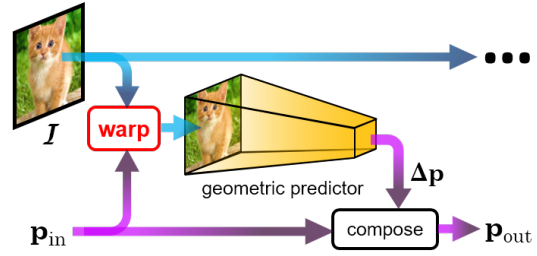

Figure 3: A learnable warping module with geometry preserved, termed as **c-STNs**. The *warp parameters* are passed through the network instead of the *warped images*.

tions are implicitly composed through multiple resampling of the images. We advocate that these transformations are able to be and should be explicitly defined and composed. Unlike the Spatial Transformer module in Fig. 1, the geometry is preserved in **p** instead of being absorbed into the output image. Furthermore, c-STNs allows repeated concatenation, illustrated in Fig. 4, where updates to the warp can be iteratively predicted. This eliminates the boundary effect because pixel information outside the cropped image is also preserved until the final transformation.

The derivative of warp compositions can also be mathematically expressed in closed forms. Consider the input and output warp parameters $\mathbf{p}_{\text{in}}$ and $\mathbf{p}_{\text{out}}$ in Fig. 3. Taking the case of affine warps for example, the parameters $\mathbf{p} = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 & p_5 & p_6 \end{bmatrix}^\top$ are relatable to transformation matrices in the homogeneous coordinates as

$$\mathbf{M}(\mathbf{p}) = \begin{bmatrix} 1+p_1 & p_2 & p_3 \\ p_4 & 1+p_5 & p_6 \\ 0 & 0 & 1 \end{bmatrix}. \quad (9)$$

From the definition of warp composition, the warp parameters are related to the transformation matrices through

$$\mathbf{M}(\mathbf{p}_{\text{out}}) = \mathbf{M}(\Delta\mathbf{p}) \cdot \mathbf{M}(\mathbf{p}_{\text{in}}). \quad (10)$$

We can thus derive the derivative to be

$$\frac{\partial\mathbf{p}_{\text{out}}}{\partial\mathbf{p}_{\text{in}}} = \mathbf{I} + \begin{bmatrix} \Delta p_1 & 0 & 0 & \Delta p_2 & 0 & 0 \\ 0 & \Delta p_1 & 0 & 0 & \Delta p_2 & 0 \\ 0 & 0 & \Delta p_1 & 0 & 0 & \Delta p_2 \\ \Delta p_4 & 0 & 0 & \Delta p_5 & 0 & 0 \\ 0 & \Delta p_4 & 0 & 0 & \Delta p_5 & 0 \\ 0 & 0 & \Delta p_4 & 0 & 0 & \Delta p_5 \end{bmatrix}$$

$$\frac{\partial\mathbf{p}_{\text{out}}}{\partial\Delta\mathbf{p}} = \mathbf{I} + \begin{bmatrix} p_{\text{in},1} & p_{\text{in},4} & 0 & 0 & 0 & 0 \\ p_{\text{in},2} & p_{\text{in},5} & 0 & 0 & 0 & 0 \\ p_{\text{in},3} & p_{\text{in},6} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & p_{\text{in},1} & p_{\text{in},4} & 0 \\ 0 & 0 & 0 & p_{\text{in},2} & p_{\text{in},5} & 0 \\ 0 & 0 & 0 & p_{\text{in},3} & p_{\text{in},6} & 0 \end{bmatrix}, \quad (11)$$
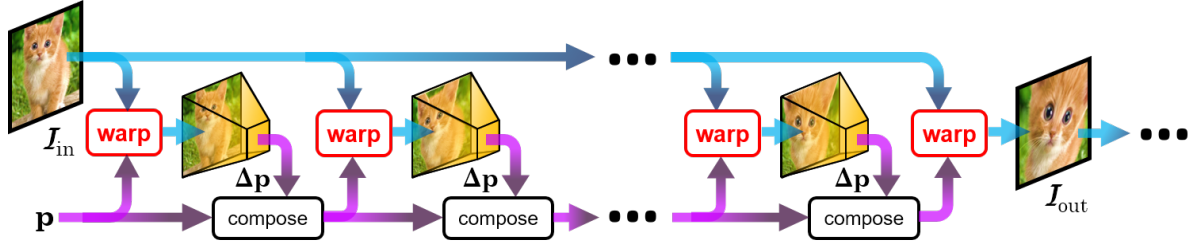
Figure 4: **Multiple concatenation of c-STNs** for an iterative alignment framework.

where $\mathbf{I}$ is the identity matrix. This allows the gradients to backpropagate into the geometric predictor.

It is interesting to note that the expression of $\frac{\partial \mathbf{P}_{\text{out}}}{\partial \mathbf{P}_{\text{in}}}$ in (11) has a very similar expression as in Residual Networks [4, 5], where the gradients contains the identity matrix $\mathbf{I}$ and "residual components". This suggests that the warp parameters from c-STNs are generally insensitive to the vanishing gradient phenomenon given the predicted warp parameters $\Delta \mathbf{p}$ is small, and that it is possible to repeat the warp/composition operation by a large number of times.

We also note that c-STNs are highly analogous to classic alignment algorithms. If each geometric predictor consists of a single linear layer, *i.e.* the appearance-geometry relationship is assumed to be linearly approximated, then it performs equivalent operations as the compositional LK algorithm. It is also related to SDM, where heuristic features such as SIFT are extracted before each regression layer. Therefore, c-STNs can be regarded as a generalization of LK and SDM, differing that the features for predicting the warp updates can be learned from data and incorporated into an end-to-end learning framework.

4.2. Recurrent Spatial Transformations

Of all variants of the LK algorithm, the IC form [2] has a very special property in that the linear regressor remains constant across iterations. The steepest descent image $\frac{\partial \mathcal{T}(\mathbf{0})}{\partial \mathbf{P}}$ in (5) is independent of the input image and the current estimate of $\mathbf{p}$; therefore, it is only needed to be computed once. In terms of model capacity, IC-LK further reduces the necessary learnable parameters compared to canonical LK, for the same regressor can be applied repeatedly and converges provided a good initialization. The main difference from canonical LK and IC-LK lies in that the warp update $\Delta \mathbf{p}$ should be compositionally applied in the inverse form. We redirect the readers to [2] for a full treatment of IC-LK, which is out of scope of this paper.

This inspires us to propose the Inverse Compositional Spatial Transformer Network (IC-STN). Fig. 5 illustrates the recurrent module of IC-STN: the warp parameters $\mathbf{p}$ is iteratively updated by $\Delta \mathbf{p}$, which is predicted from the current warped image with the same geometric predictors. This
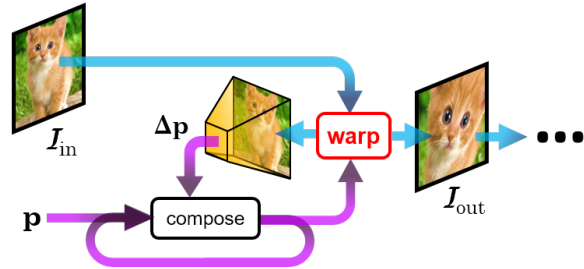

Figure 5: Illustration of the proposed **Inverse Compositional Spatial Transformer Network** (IC-STN). The same geometric predictor is learned to predict recurrent spatial transformations that are composed together to warp the input image.

allows one to recurrently predict spatial transformations on the input image. It is possible due to the close spatial proximity of pixel intensities within natural images: there exists high correlation between pixels in close distances.

In the IC-LK algorithm, the predicted warp parameters are inversely composed. Since the IC-STN geometric predictor is optimized in an end-to-end learning framework, we can absorb the inversion operation into the geometric predictor without explicitly defining it; in other words, IC-STNs are able to directly predict the inverse parameters. In our experiments, we find that there is negligible difference to explicitly perform an additional inverse operation on the predicted forward parameters, and that implicitly predicting the inverse parameters fits more elegantly in an end-to-end learning framework using backpropagation. We name our proposed method Inverse Compositional nevertheless as IC-LK is where our inspirations are drawn from.

In practice, IC-STNs can be trained by unfolding the architecture in Fig. 5 multiple times into the form of c-STNs (Fig. 4), sharing the learnable parameters across all geometric predictors, and backpropagating the gradients as described in Sec. 4.1. This results in a single effective geometric predictor that can be applied multiple times before performing the final warp operation that suits subsequent tasks such as classification.
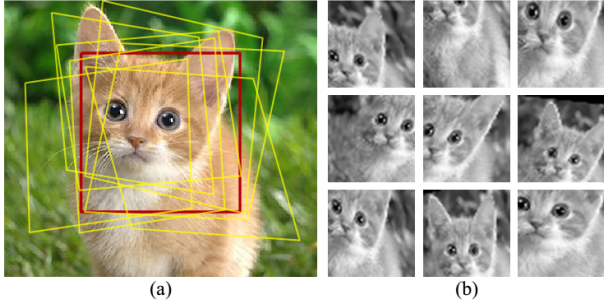
Figure 6: Visualization of the image and perturbed training samples for the planar image alignment experiment. (a) Original image, where the red box indicates the ground-truth warp and the yellow boxes indicate example generated warps. (b) Examples of the perturbed images (affine warps with $\sigma = 7.5$ in this case).

Model	$\sigma = 2.5$	$\sigma = 5$	$\sigma = 7.5$	$\sigma = 10$
c-STN-1	2.699	5.576	9.491	9.218
IC-STN-2	0.615	2.268	5.283	5.502
IC-STN-3	0.434	1.092	2.877	3.020
IC-STN-4	0.292	0.481	1.476	2.287
IC-STN-6	0.027	0.125	0.245	1.305

Table 1: **Test error for the planar image alignment experiment** under different extents of initial perturbations. The number following the model names indicate the number of warp operations unfolded from IC-STN during training.

5. Experiments

5.1. Planar Image Alignment

To start with, we explore the efficacy of IC-STN for planar alignment of a single image. We took an example image from the Caffe library [8] and generated perturbed images with affine warps around the hand-labeled ground truth, shown in Fig. 6. We used image samples of size 50×50 pixels. The perturbed boxes are generated by adding i.i.d. Gaussian noise of standard deviation σ (in pixels) to the four corners of the ground-truth box plus an additional translational noise from the same Gaussian distribution, and finally fitting the box to the initial warp parameters **p**.

To demonstrate the effectiveness of iterative alignment under different amount of noise, we consider IC-STNs that consist of a single learnable linear layer with different numbers of learned recurrent transformations. We optimize all networks in terms of L_2 error between warp parameters with stochastic gradient descent and a batch size of 100 perturbed training samples generated on the fly.

The test error is illustrated in Table 1. We see from c-STN-1 (which is equivalent to IC-STN-1 with only one warp operation unfolded) that a single geometric warp pre-
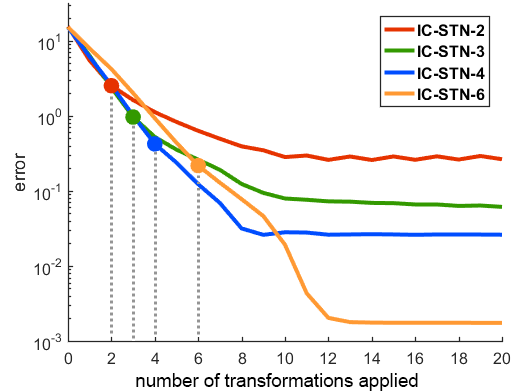

Figure 7: **Evaluation on trained IC-STNs**, where the dot on each curve corresponds to the number of recurrent transformations unfolded during training.

dictor has limited ability to directly predict the optimal geometric transformation. Reusing the geometric predictor to incorporating multiple spatial transformations yields better alignment performance given the same model capacity.

Fig. 7 shows the test error over the number of warp operations applied to the learned alignment module. We can see that even when the recurrent spatial transformation is applied more times than trained with, the error continues to decrease until some of point of saturation, which typically does not hold true for classical recurrent neural networks. This implies that IC-STN is able to capture the correlation between appearance and geometry to perform gradient descent on a learned cost surface for successful alignment.

5.2. MNIST Classification

In this section, we demonstrate how IC-STNs can be utilized in joint alignment/classfication tasks. We choose the MNIST handwritten digit dataset [10], and we use a homography warp noise model to perturb the four corners of the image and translate them with Gaussian noise, both with a standard deviation of 3.5 pixels. We train all networks for 200K iterations with a batch size of 100 perturbed samples generated on the fly. We choose a constant learning rate of 10^{-2} for the classification networks and 10^{-4} for the geometric predictors as we find the geometric predictor sensitive to large changes. We evaluate the classification accuracy on the test set using the same warp noise model.

We compare IC-STN to several network architectures, including a baseline CNN with no spatial transformations, the original STN from Jaderberg *et al.*, and c-STNs. All networks with spatial transformations employ the same classification network. The results as well as the architectural details are listed in Table 2. We can see that classical CNNs do not handle large spatial variations efficiently with data augmentation. In the case where the digits may be occluded,

Model	Test error	Capacity	Architecture
CNN(a)	6.597 %	39079	conv(3×3, 3)-conv(3×3, 6)-P-conv(3×3, 9)-conv(3×3, 12)-FC(48)-FC(10)
STN(a)	4.944 %	39048	[conv(7×7, 4)-conv(7×7, 8)-P-FC(48)-FC(8)]×**1** → conv(9×9, 3)-FC(10)
c-STN-1(a)	3.687 %	39048	[conv(7×7, 4)-conv(7×7, 8)-P-FC(48)-FC(8)]×**1** → conv(9×9, 3)-FC(10)
c-STN-2(a)	2.060 %	38528	[conv(9×9, 4)-FC(8)]×**2** → conv(9×9, 3)-FC(10)
c-STN-4(a)	1.476 %	37376	[FC(8)]×**4** → conv(9×9, 3)-FC(10)
IC-STN-2(a)	**1.905** %	39048	[conv(7×7, 4)-conv(7×7, 8)-P-FC(48)-FC(8)]×**2** → conv(9×9, 3)-FC(10)
IC-STN-4(a)	**1.230** %	39048	[conv(7×7, 4)-conv(7×7, 8)-P-FC(48)-FC(8)]×**4** → conv(9×9, 3)-FC(10)
CNN(b)	19.065 %	19610	conv(9×9, 2)-conv(9×9, 4)-FC(32)-FC(10)
STN(b)	9.325 %	18536	[FC(8)]×**1** → conv(9×9, 3)-FC(10)
c-STN-1(b)	8.545 %	18536	[FC(8)]×**1** → conv(9×9, 3)-FC(10)
IC-STN-2(b)	**3.717** %	18536	[FC(8)]×**2** → conv(9×9, 3)-FC(10)
IC-STN-4(b)	**1.703** %	18536	[FC(8)]×**4** → conv(9×9, 3)-FC(10)

Table 2: **Classification error on the perturbed MNIST test set.** The non-recurrent networks have similar numbers of layers and learnable parameters but different numbers of warp operations (bold-faced). The filter dimensions are shown in parentheses, where those of the geometric predictor(s) are in green and those of the subsequent classification network are in blue (P denotes a 2×2 max-pooling operation). Best viewed in color.

Figure 8: **Sample alignment results of IC-STN-4(a)** on the MNIST test set with homography warp perturbations. The first row of each column shows the initial perturbation; the middle three rows illustrates the alignment process (iterations 1 to 3); the second last row shows the final alignment before feeding into the classification network. The last row shows the alignment from the original STN: the cropped digits are the results of the boundary effect.

however, trading off capacity for a single deep predictor of geometric transformation also results in poor performance. Incorporating multiple transformers lead to a significant improvement in classification accuracy; further comparing c-STN-4(a) and IC-STN-4(b), we see that IC-STNs are able to trade little accuracy off for a large reduction of capacity compared to its non-recurrent counterpart.

Fig. 8 shows how IC-STNs learns alignment for classification. In many cases where the handwritten digits are occluded, IC-STN is able to automatically warp the image and reveal the occluded information from the original image. There also exists smooth transitions during the alignment, which confirms with the recurrent spatial transformation concept IC-STN learns. Furthermore, we see that the outcome of the original STN becomes cropped digits due to the boundary effect described in Sec. 4.1.

We also visualize the overall final alignment performance by taking the mean and variance on the test set appearance before classification, shown in Fig. 9. The

mean/variance results of the original STN becomes a down-scaled version of the original digits, reducing information necessary for better classification. From c-STN-1, we see that a single geometric predictor is poor in directly predicting geometric transformations. The variance among all aligned samples is dramatically decreased when more warp operations are introduced in IC-STN. These results support the fact that elimination of spatial variations within data is crucial to boosting the performance of subsequent tasks.

5.3. Traffic Sign Classification

Here, we show how IC-STNs can be applied to real-world classification problems such as traffic sign recognition. We evaluate our proposed method with the German Traffic Sign Recognition Benchmark [18], which consists of 39,209 training and 12,630 test images from 43 classes taken under various conditions. We consider this as a challenging task since many of the images are taken with motion blurs and/or of resolution as low as 15×15 pixels. We

Model	Test error	Capacity	Architecture
CNN	8.287 %	200207	conv(7×7, 6)-conv(7×7, 12)-P-conv(7×7, 24)-FC(200)-FC(43)
STN	6.495 %	197343	[conv(7×7, 6)-conv(7×7, 24)-FC(8)]×**1** → conv(7×7, 6)-conv(7×7, 12)-P-FC(43)
c-STN-1	5.011 %	197343	[conv(7×7, 6)-conv(7×7, 24)-FC(8)]×**1** → conv(7×7, 6)-conv(7×7, 12)-P-FC(43)
IC-STN-2	4.122 %	197343	[conv(7×7, 6)-conv(7×7, 24)-FC(8)]×**2** → conv(7×7, 6)-conv(7×7, 12)-P-FC(43)
IC-STN-4	**3.184** %	197343	[conv(7×7, 6)-conv(7×7, 24)-FC(8)]×**4** → conv(7×7, 6)-conv(7×7, 12)-P-FC(43)

Table 3: **Classification error on the perturbed GTSRB test set**. The architectural descriptions follow that in Table 2.
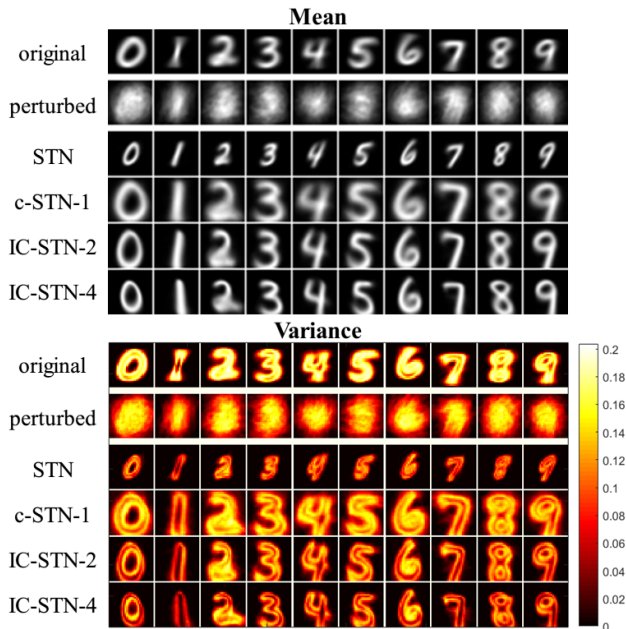

Figure 9: **Mean/variance of the aligned appearances** from the 10 classes of the test set (homography perturbations).

Figure 10: **Sample alignment results of IC-STN-4** on the GTSRB test set in comparison to the original STN.

rescale all images and generate perturbed samples of size 36×36 pixels with the same homography warp noise model described in Sec. 5.2. The learning rate is 10^{-3} for the classification networks and 10^{-5} for the geometric predictors.

We set the controlled model capacities to around 200K learnable parameters and perform similar comparisons to the MNIST experiment. Table 3 shows the classification error on the perturbed GTSRB test set. Once again, we see a considerable amount of classification improvement of IC-STN from learning to reuse the same geometric predictor.
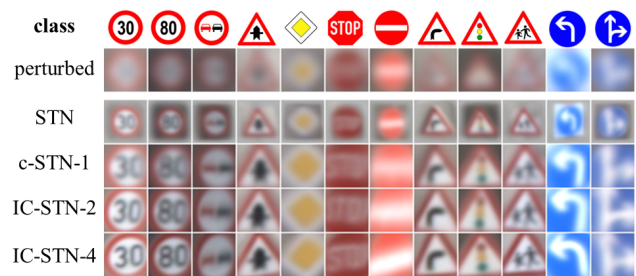

Figure 11: **Mean aligned appearances** for classification from sampled classes of the GTSRB test set.

Fig. 10 compares the aligned images from IC-STN and the original STN before the classification networks. Again, IC-STNs are able to recover occluded appearances from the input image. Although STN still attempts to center the perturbed images, the missing information from occlusion degrades its subsequent classification performance.

We also visualize the aligned mean appearances from each network in Fig. 11, and it can be observed that the mean appearance of IC-STN becomes sharper as the number of warp operations increase, once again indicating that good alignment is crucial to the subsequent target tasks. It is also interesting to note that not all traffic signs are aligned to be fit exactly inside the bounding boxes, *e.g.* the networks finds the optimal alignment for stop signs to be zoomed-in images while excluding the background information outside the octagonal shapes. This suggests that in certain cases, only the pixel information inside the sign shapes are necessary to achieve good alignment for classification.

6. Conclusion

In this paper, we theoretically connect the core idea of the Lucas & Kanade algorithm with Spatial Transformer Networks. We show that geometric variations within data can be eliminated more efficiently through multiple spatial transformations within an alignment framework. We propose Inverse Compositional Spatial Transformer Networks for predicting recurrent spatial transformations and demonstrate superior alignment and classification results compared to baseline CNNs and the original STN.

References

[1] A. Asthana, S. Zafeiriou, S. Cheng, and M. Pantic. Incremental face alignment in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1859–1866, 2014. 3

[2] S. Baker and I. Matthews. Lucas-kanade 20 years on: A unifying framework. *International journal of computer vision*, 56(3):221–255, 2004. 1, 2, 5

[3] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893. IEEE, 2005. 3

[4] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015. 1, 5

[5] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. *arXiv preprint arXiv:1603.05027*, 2016. 5

[6] G. Huang, M. Mattar, H. Lee, and E. G. Learned-miller. Learning to align from scratch. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 764–772. Curran Associates, Inc., 2012. 1

[7] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In *Advances in Neural Information Processing Systems*, pages 2017–2025, 2015. 1, 3

[8] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678. ACM, 2014. 6

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 1

[10] Y. LeCun, C. Cortes, and C. J. Burges. The mnist database of handwritten digits, 1998. 6

[11] C.-H. Lin, R. Zhu, and S. Lucey. The conditional lucas & kanade algorithm. In *European Conference on Computer Vision (ECCV)*, pages 793–808. Springer International Publishing, 2016. 3

[12] J. L. Long, N. Zhang, and T. Darrell. Do convnets learn correspondence? In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 1601–1609. Curran Associates, Inc., 2014. 1

[13] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004. 3

[14] B. D. Lucas, T. Kanade, et al. An iterative image registration technique with an application to stereo vision. In *IJCAI*, volume 81, pages 674–679, 1981. 1, 2

[15] S. Ren, X. Cao, Y. Wei, and J. Sun. Face alignment at 3000 fps via regressing local binary features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1685–1692, 2014. 3

[16] P. Y. Simard, D. Steinkraus, and J. C. Platt. Best practices for convolutional neural networks applied to visual document analysis. In *ICDAR*, volume 3, pages 958–962, 2003. 1

[17] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 1

[18] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. The german traffic sign recognition benchmark: a multi-class classification competition. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*, pages 1453–1460. IEEE, 2011. 7

[19] X. Xiong and F. De la Torre. Supervised descent method and its applications to face alignment. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 532–539, 2013. 3