# Revisiting Stochastic Completion Fields

Ryan Marten, Sid Gupta, Chandra Gummaluru
Department of Computer Science, University of Toronto

## Abstract

We revisit Stochastic Completion Fields: A Neural Model for Illusory Contour Shape and Salience, published by Lance R. Williams and David W. Jacobs in 1997. First, we formally present the mathematical theory as described by Williams and Jacobs, which is centered around the assumption that the prior probability distribution of completion curves can be modelled by random walks of a particle. Then, we explain our Python implementation of the Stochastic Completion Fields theory, filling in missing details about the algorithm that were left out of the original paper. We justify our implementation decisions with the ultimate goal of computational efficiency. We reproduce Williams and Jacobs' experimental results and demonstrate the ability of our implementation to find contour completions in illusions that are consistent with human perception. We then further analyze the behaviour of the algorithm by experimenting with hyperparameters: the count, length, and variance of the random walks.

## 1.1 Introduction

In the original 1997 paper "Stochastic Completion Fields: A Neural Model of Illusory Contour Shape and Salience", Williams and Jacobs provided a new theory for shape completion [1]. The shape completion problem is described by Williams as "computing the shape and relative likelihood of the family of curves which potentially connect a set of boundary fragments". In other words: recovering missing edges to complete a figure's shape. Early Computer Vision scientists were particularly focused on shape analysis and perceptual grouping. Williams and Jacobs wrote "It is widely acknowledged that perceptual organization (i.e. segmentation, grouping) is among the most difficult problems facing researchers in computer vision today" [1]. Today, it still remains a difficult unsolved problem, but is less popular in contrast to fully end-to-end deep learning systems that perform higher level Computer Vision tasks.

We return to stochastic completion fields and reconsider how the model can be used today. The shape completion conducted by the stochastic completion fields method could provide an important intermediate step in a principled computer vision pipeline that disentangles the low, middle, and high level vision computation. Natural images are often represented as line drawings to reduce the dimensionality and complexity of the data by summarizing shape information with meaningful, complete contours [2]. Due to factors like lighting and surface discontinuities, typical edge detection can often create broken contours [3]. Additionally, edge detection alone can not account for factors like occlusion.

In this project, we follow Williams and Jacobs in using a series of popular illusions from the visual psychology literature to demonstrate contour completion. This task acts as a motivating challenge for modeling human perception, requiring adherence to the Gestalt Laws of Proximity, Continuity, and Closure [4]. A model with these properties could help create more robust line drawings to describe the shape of objects in an image. In other words, we view stochastic completion fields as a tool to recover missing contours that are essential for the understanding of the shapes in an image.

## 1.2 Additional Background

The motivation behind computationally solving illusory contour completion is captured in the earliest work on the topic by Ullman in 1976, who argues that illusions give meaningful insights into the inner workings of the human visual system that can be used to design computer vision systems [5]. He proposes that the neural mechanism that fills in illusory contours might play an active role in all low level processing done by the human visual system and not only triggered in special cases. By building a system that can mimic the human behaviour of illusory contour completion, we gain an important step in a computer vision pipeline to understand shapes in real world images. This is especially important when one considers the imperfections of real world scenes where many object boundaries are occluded and must be inferred in much the same way as illusory contours.

What differentiated the stochastic completion fields paper from the illusory shape completion literature at the time was the major assumption that contour completions between boundary fragments could be modeled as random walks of a particle. Other work employed strategies that approached shape completion in terms of co-circularity and minimizing curve energy [6, 7]. A major goal for all these solutions was to create an algorithm that was neurologically plausible, given the current understanding of the human visual system, and fast. The theory of stochastic completion fields was proposed to meet both of those goals. Instrumentally, Williams and Jacobs make an additional key assumption that the motion of the random walk depends only on the orientation and position of the particle at the previous time step, describing the random walk as a Markov process. This allows them to compute the final completion field efficiently by the simple multiplication of a source and sink field. Williams and Jacobs then discuss how a neural system could be theoretically created to carry out this computation.

Williams and Jacobs also deviated from the norm at the time by proposing that a distribution of curves, rather than only a single completion be computed. Instead of only considering the maximum likelihood completion, Williams and Jacobs argued that the statistics of the distribution could yield additional information on the completion characteristics. Namely, the mode, magnitude, and variance of the Stochastic Completion Field represented the shape, salience, and sharpness of the perceived contour. In visualizing the final completion fields, this relationship is apparent.

Following the initial introduction of stochastic completion fields, a handful of variations on the theory have been proposed [8,9,10,11,12]. Williams and Jacobs themselves published a paper immediately after the original, describing the local parallel computation of stochastic completion fields, which results in a more plausible neural model that focuses on local computation, and faster computation [8]. In 1999, Williams introduced a method that computes stochastic completion fields using a resolution pyramid and is able to achieve a linear-time complexity that drastically decreases computational costs, reducing experiments from running 1 hour long to 2 minutes [9]. Shiyan discusses a method that uses transition probabilities instead of Monte Carlo simulation of random walks to generate a stochastic completion field [12]. Quan implements the local parallel method and discusses it’s failures [11].

We revisit the original paper to reimplement the theoretical framework into an efficient algorithm built with a modern programming language and its packages. We make some novel contributions that allows us to compute the Stochastic Completion Field with complexity competitive with the later variations, but without sacrificing any of the original intended behaviour.

## 1.3 Stochastic Completion Fields Definition

Our goal in re-presenting the mathematical theory of stochastic completion fields is to be more clear and comprehensive. The main goal of stochastic completion fields is that we fill missing edges by connecting boundary fragments. These boundary fragments end-points are represented as "source" or "sink" points in the model. Our goal is to find the distribution of likely curves that travel from the source to the sink. In Figure 1. is an example of what we define as a source and sink and a possible completion.

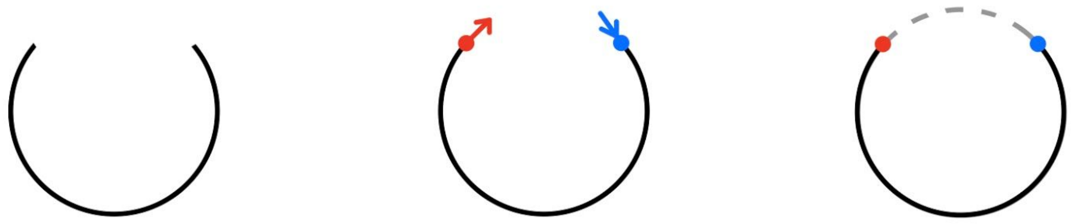

*Figure 1 - Left: Boundary Fragments; Center: Source and Sink; Right: Possible Completion*

Williams and Jacobs do not provide a theory on how to recover the source and sink points from intensities of image pixels. In their experiments section, they suggest using steerable filters, but leave a more rigorous discussion to further research work. For our purposes, we too will consider the source and sink as known inputs. It is important to note that sources and sinks have a location and an orientation.

The main contribution of stochastic completion fields is modeling completion curves as random walks. Random walks are defined as a particle in motion, randomly changing its direction at every time step, moving a unit length at each time step. Figure 2 depicts an example random walk. Not only does this model provide the desirable properties of creating continuous curves locally in the area of the starting location (Gestalt principle of proximity and continuity), it also allows us to simplify computation.

Since the random walk location and orientation only depends on the previous time step and location and orientation, we can use the Markov assumption to factor out the source and sink fields from the computation of the completion field. This works because we can consider the journey from the source to the sink as two independent paths from the source to an intermediate point, and from the intermediate point to the sink.

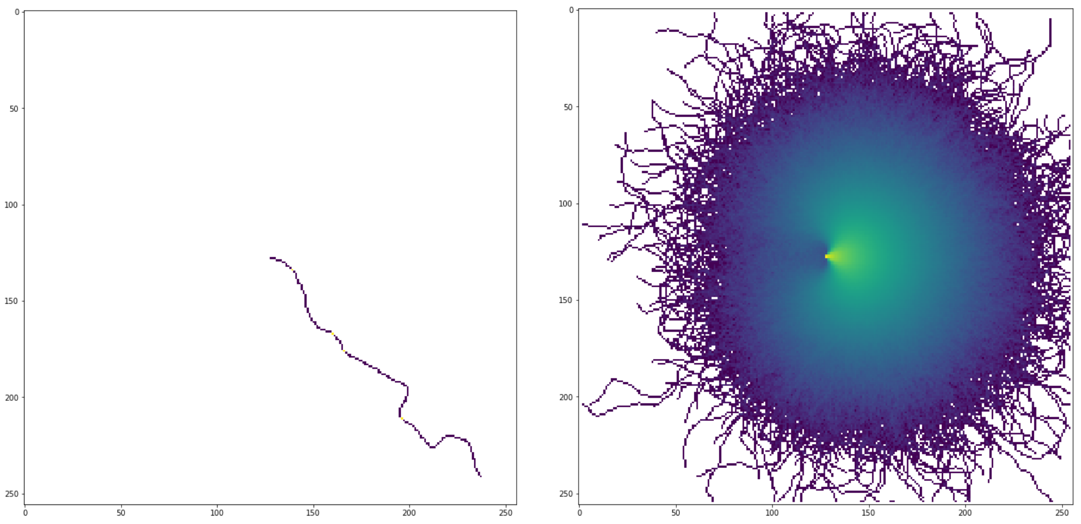

*Figure 2 - Left: A single random walk from the center; Right: 1,00,000 random walks (logarithmic scale)*

We now formally describe the process of computing the completion field. We begin by considering a particle on a random walk in the 2-dimensional plane, i.e., $\mathbb{R}^2$.

The state, $s(t)$ of the particle at some time, $t$, is actually defined by both its position, $(x(t), y(t))$, and orientation $\theta(t)$, that is, $s(t) = (x(t), y(t), \theta(t))$. As a convention, we will assume $\theta \in [-\pi, \pi)$. For this reason, it is more convenient to view the random walk as occurring in a 3-dimensional space, namely, $\mathbb{R}^2 \times [-\pi, \pi)$.

We assume the particle always moves with a unit velocity along its current orientation, $\theta(t)$, that is, $\dot{x} = \cos\theta, \dot{y} = \sin\theta$. The orientation changes randomly at each time-step. More precisely, the angular velocity is a normally distributed random variable with zero mean, and a variance of $\sigma^2$, i.e., $\dot{\theta} \equiv \mathcal{N}(0, \sigma^2; t)$. As a result, $x(t)$ and $y(t)$, in addition to $\theta(t)$, are also random variables.

Let $p(u, s, v)$ be the probability density that a random walk begins at some state, $u$, and passes through some intermediate state, $s = (x_s, y_s, \theta_s)$ before ending at some other state, $v$.

We seek the probability density that a random walk begins at *any* source, passes through the intermediate state $s$ and ends at *any* sink, that is,

$$c(s) = \iint p(u, s, v) du dv$$

Here $c(s)$ is what we refer to as a *completion field*.

Let $p_U(u)$ denote the probability density of $u$ being a *source*, that is, the probability density that a random walk begins at $u$.

Given a set of *source states*, $U$, one way to define $p_U$ is

$$p_U(u) = \frac{1}{|U|} \sum_{\forall u' \in U} \delta(u - u'),$$

that is, every source is equally likely.

Similarly, if $p_V(v)$ is the probability density of $v$ being a sink, that is, the probability density that a random walk ends at $v$, one can define

$$p_V(v) = \frac{1}{|V|} \sum_{\forall v' \in V} \delta(v - v'),$$

where $V$ is a set of *sink states*.

It is worth pointing out that there are of-course, other ways to define $p_U$ and $p_V$, however, we do not consider them here.

In any case though, $p(u, s, v)$ can be decomposed into four probabilities, namely:
- that $u$ is a source, i.e., $p_U(u)$,
- that a random walk beginning at $u$, ends at $s$, i.e., $p(s|u)$,
- that a random walk beginning at $u$ and passing through $s$, ends at $v$, i.e., $p(v|u, s)$, and
- the probability that $v$ is a sink, i.e., $p_V(v)$.

In other words, we may write,

$$p(u, s, v) = p_U(u)p(s|u)p(v|u, s)p_V(v).$$

However, the random walk we have described is a Markov process; that is, the probability distribution of the next state, $(x', y', \theta')$, given the current state, $(x, y, \theta)$, is independent of any previous states. We have $x' = x + \cos(\theta')dt$, $y' = y + \sin(\theta')dt$ and $\theta' = \theta + \dot{\theta}dt$, where $\dot{\theta}$, is random but invariant to state.

$$p(u, s, v) = p_U(u)p(s|u)p(v|s)p_V(v)$$

Thus, we may rewrite

$$c(s) = \iint p_U(u)p(s|u)p(v|s)p_V(v)dudv$$
$$= \int p_U(u)p(s|u)du \int p_V(v)p(v|s)dv.$$

If we define the *source field* as

$$p'_U(s) = \int p_U(u)p(s|u)du,$$

and *sink field* as

$$p'_V(s) = \int p_V(v)p(v|s)dv,$$

then the completion field is the product of said source and sink fields, i.e., $c(s) = p'_U(s)p'_V(s)$.

Since $p_U$ and $p_V$ are given, all that remains is to compute $p(s|u)$ and $p(v|s)$.

We first consider $p(s|u)$. It turns out that $p(s|u)$ has an analytic expression, namely, a Green's function. However, for our purposes, we resort to approximating it via a monte-carlo approach. More precisely, we can approximate $p(s|u)$ by simulating $n$ random walks beginning in the state $u$, and considering the fraction of such walks that also end in the state $s$. For large enough $n$, this approximation is sufficient.

Since $\dot{\theta}$ is invariant to translation and rotation, so too is $p(s|u)$. Therefore, if $u = (x_u, y_u, \theta_u)$ and $s = (x_s, y_s, \theta_s)$, we may write

$$p(s|u) = p(s'|(0,0,0))$$

where $x_{s'} = (x_s - x_u)\cos\theta_u + (y_s - y_u)\sin\theta_u$, $y_{s'} = -(x_s - x_u)\sin\theta_u + (y_s - y_u)\cos\theta_u$, and, $\theta_{s'} = \theta_s - \theta_u$. The upshot of this equality is that we need only compute the monte-carlo simulation assuming the source is the origin and then transform (rotate and translate) the resulting distribution for each source, $u$, instead of actually performing the monte-carlo simulation for every source.

We now consider $p(v|s)$. As we did with $p(s|u)$, we can analogously show that $p(v|s) = p(v'|(0,0,0))$, though this is less useful, since we would still need to transform the distribution for every possible state, $s$. However, the rotation and translation invariance of $p(v|s)$ can be used to also show that

$$p(v|s) = p(s^*|v^*)$$

where $s^* = (x_s, y_s, \theta_s + \pi)$ is the *conjugate* of the state, $s$.

Intuitively, the equation above says that the probability of a random walk occurring is equivalent to the probability that the reverse walk occurs. As far as we are aware, this property has not been exploited in previous works. The implication is that we need only transform the distribution for every sink, $v$, (more precisely, the conjugate of the sink, $v^*$) instead of every possible state, $s$.

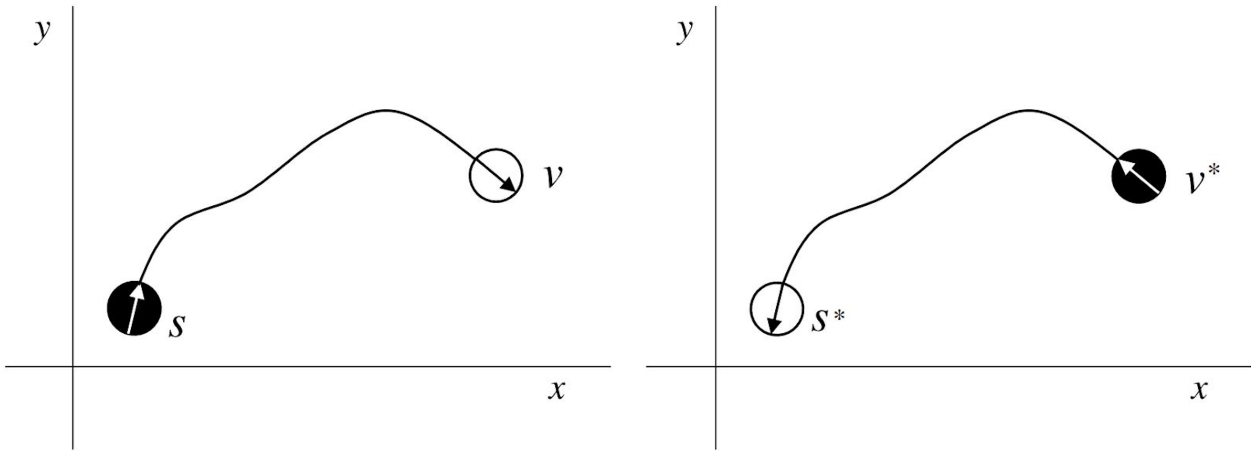

*Figure 3 - Completing the random walk in reverse is equivalent to rotating all state orientations by π*

## 2. Methodology and Implementation Decisions

Our implementation uses Python and leverages the convenience of libraries such as numpy and matplotlib for calculation and visualization. The full code is located on github, but here we highlight particular implementation decisions that are key to archiving our goal of computational efficiency.

We first generate the set of random walks to model the prior distributions of the source and sink fields. We propose generating the Monte Carlo simulation of random walks once and reusing it for all future experiments. Effectively, the cost of computing the walks is amortized over all future completion fields for any set of source and sinks in any image, given constant parameters $\tau$ and *diffusivity*. For our experiments we generate 1,000,000 random walks to represent a source / sink field located at the origin with the orientation $\theta = 0$. The random walks are first generated in a real-value space, then discretized to an integer lattice, representing the locations of pixels in the image. In the real-value representation, walks are stored as a simple list of states, which are 3-tuples containing $(x, y, \theta)$. In the discretized representation, the distribution is stored as an order-3 tensor of size (256, 256, 36) for an image of size (256,256) with 36 orientation bins. In the discretization step, we make a key choice to assign probability values by essentially calculating a histogram of unique states and their counts and assigning these counts into the tensor. Compared to iterating through the walks and incrementing the counts one by one, this approach is several orders of magnitude faster.

We then compute our source and sink fields by rotating and translating the random walks matrix to the sources' and sinks' location and orientation. Using homogeneous coordinates, we can define the rotation and translation as an efficient matrix multiplication.

$$R(d\theta) = \begin{bmatrix} \cos d\theta & \sin d\theta & 0 & 0 \\ -\sin d\theta & \cos d\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, T(dx, dy, d\theta) = \begin{bmatrix} 1 & 0 & 0 & dx \\ 0 & 1 & 0 & dy \\ 0 & 0 & 1 & d\theta \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T(dx, dy, d\theta)R(d\theta)\begin{bmatrix} P \\ 1^{\top} \end{bmatrix}.$$

We are able to compute the sink field in the exact same manner as the sink field because we make a key observation: the probability of a path from one point to another is equivalent in reflection, considering the same points, but with conjugate angles. This is described above as $p(v|s) = p(s^*|v^*)$.

Finally, we compute the stochastic completion field by simply multiplying the source and sink field. Overall, we were able to drastically improve the runtime of the stochastic completion algorithm from our initial attempts. The decisions we highlighted here were novel or missing from the paper and allowed us to make the original stochastic completion fields algorithm competitive with the computational complexity of the improved variations on the theory that were proposed later.

## 3.1 Experiments and Results

In addition to being efficient, our algorithm performs well on a diverse set of images. We will first illustrate a Stochastic Completion Field on the Ehrenstein figure (Figure 3):

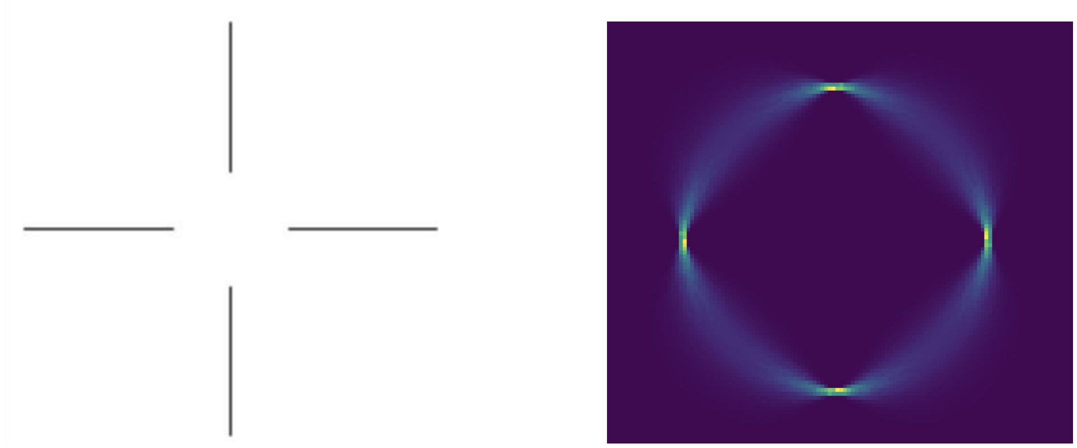

*Figure 3 - Left: Ehrenstein Figure; Right: Stochastic Completion Field of Ehrenstein Figure*

To generate the above image, we placed source and sinks at each of the four corners, going opposite directions to each other (Figure 4):

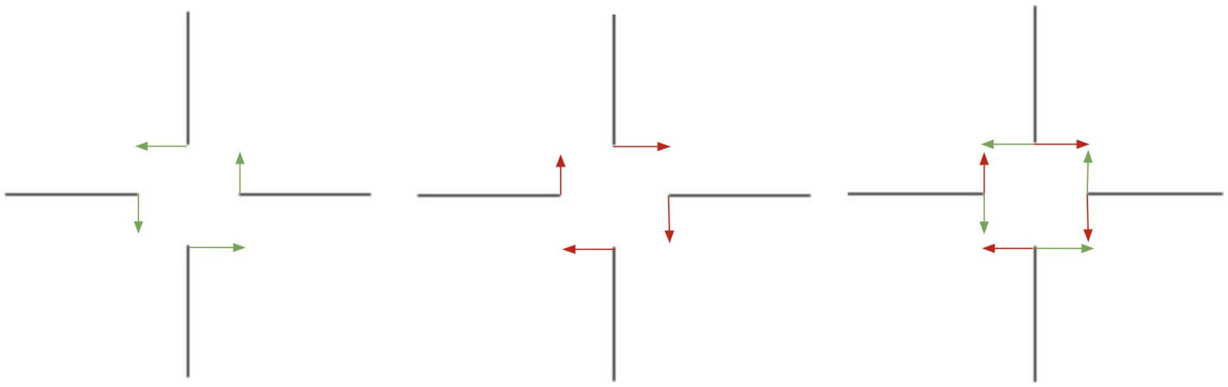

*Figure 4 - Left: Source points; Middle: Sink points; Right: Source and Sink points*

The reason why these positions were chosen is because a source/sink matching should represent a probable connection. There are four segments to the circle, and each segment is connected by a source and sink. They point perpendicular to one another, as the most probable method to walk from source to sink in that arrangement is an arc-like path.

Next we will present two more examples: The Paisley's image and the Kanizsa Triangle (Figure 5)

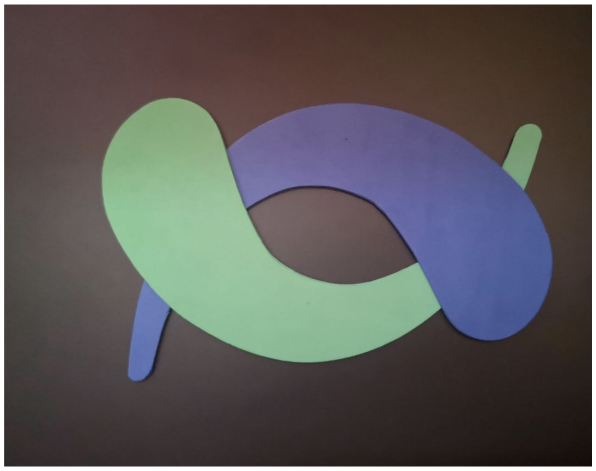
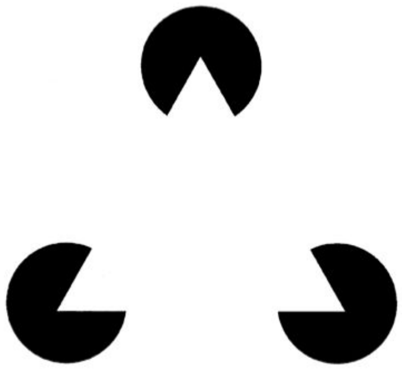

*Figure 5 - Left: Paisleys image; Right: Kanizsa Triangle*

The reason why these two images are chosen is because they each represent a perceptive phenomena that computer vision algorithms struggle to capture. As shown, the Paisleys image is represented by two cardboard cutouts overtop of each other. When one object overlaps another in an image, we call this *occlusion*. In occlusion, the covered object loses information. However, humans can easily fill in this information, and have a strong guess to what the covered part is. This cognitive function is explained by the Gestalt Law of Continuation, and we use stochastic completion fields to give computers the same kind of insight. Below we present an edge contour of the Paisleys image using a standard edge-detection algorithm, and the Stochastic Completion Field of the image. (Figure 6):

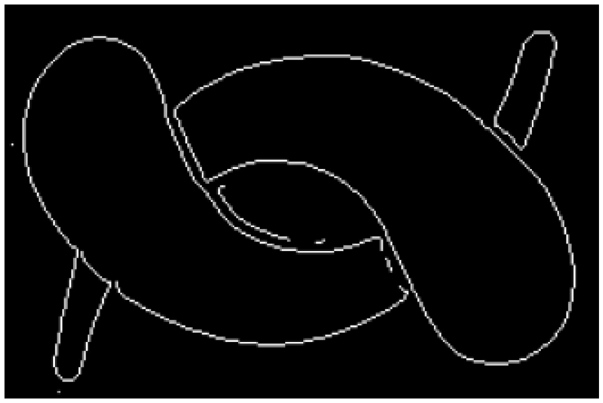
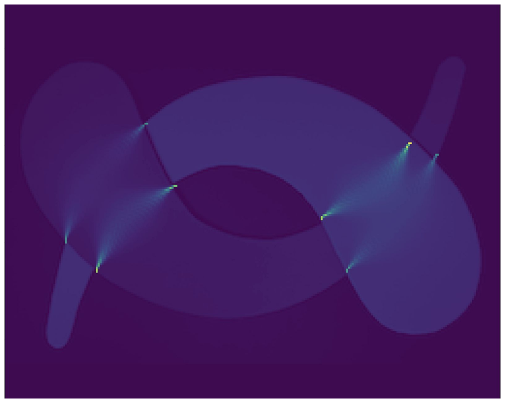

*Figure 6 - Left: Edge detection on Paisleys; Right: SCF on Paisleys*

With regards to edge detection, it looks like there are four objects. That's because the computer only examines the visible edges, and completely disregards the occluded ones.

The Stochastic Completion Field accurately conveys what the covered regions would look like, and adds information to the individual objects. That is, a computer now has insight on what the occluded edges look like. Finally, we will combine the two methodologies together (Figure 7):

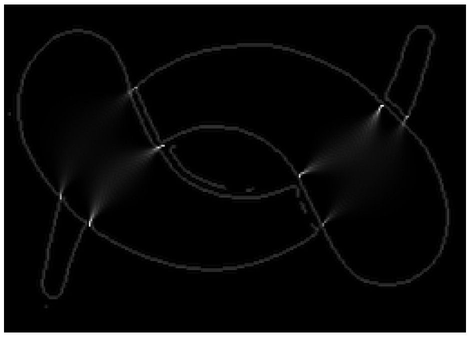
*Figure 7 - Edge detection + SCF on Paisleys*

Here we can see that edge detection is empowered to not only look at visible edges, but also the occluded ones; making a much more accurate representation of what's going on in the image. Modern edge detection is used in many applications, and with the help of stochastic completion fields, it can be augmented in a powerful way to uncover hidden information. When we (as humans) look at the Kanizsa triangle, we indeed see a white triangle. However, the information present shows that there are three black partial-circles. The Gestalt Laws of Continuation and Completion can explain why we see a triangle, but a computational model likely can't make that kind of inference.

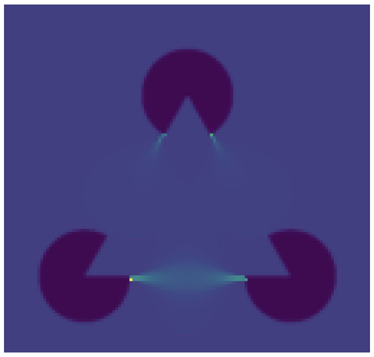
*Figure 8 - SCF on Kanizsa Triangle*

We can mark the points of ambiguity as sources and sinks, and have the computer complete the image  a similar way that we (as humans) would. In this way, the image is completed, and the computer has more information to see the triangle in this illusion. Note that Figure 8 does indeed form a complete contour; the sides of the triangle are just much lighter than the bottom, since the bottom has a more probable field due to the assignment of hyperparameters (discussed in section 3.3).

## 3.2 Experiment implementation details:

Edges are defined to be a change in gradient in the image, and the gradient of an image is perpendicular to an edge (Marr, Hildreth 1980). The principled angle at every source and sink should equal the slope of the tangent at that point in the edge. From the theorem proved by Marr and Hildreth, we can compute this tangent as the angle perpendicular to the gradient at that point. A visualization of the angles can be seen in Figure 9:

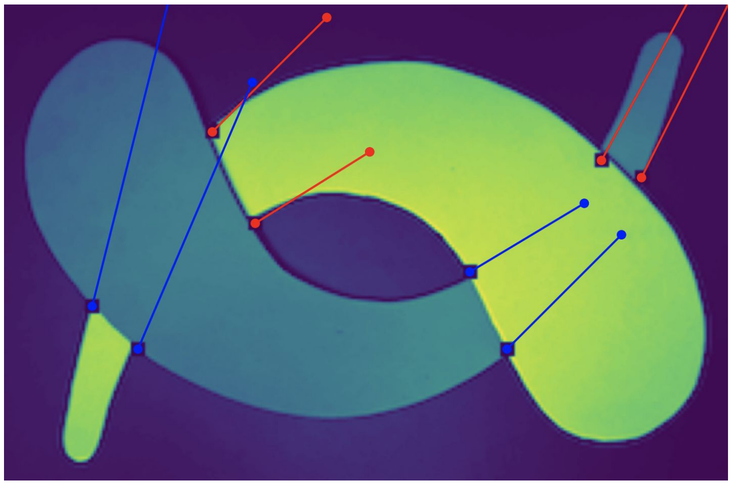

*Figure 9 - Sources and Sinks on Paisleys image, with angles*

The sources have blue lines coming out of them, and the sinks have red lines coming out of them. The angle of the source should be in the direction that goes into the sink. The angle of the sink, however, holds a direction that goes outwards rather than inwards. This is because we are moving source to sink.

## 3.3 Experiment hyperparameters:

In our implementation, we work with two main hyperparameters: Tau and Diffusivity. For each random walk, Tau controls its length, and diffusivity controls its variance. In our code, we sample from a binomial distribution with a parameter of Tau. That is, every Tau steps we take in a walk, we reduce the probability of continuing by half. At every step, we choose an angle to move in from a normal distribution, with variance = diffusivity. As a result, a larger diffusivity will result in more jagged walks, because we will pick more varied angles at every step.

How these hyperparameters should be tweaked depends mostly on the use case. A lower diffusivity is better at detecting less varied, more straight paths; whereas a higher diffusivity is better at detecting more obscure paths. In addition, because a high-variance path can look very jagged, it can cover less horizontal and vertical distance. As a result, if a high diffusivity is used, it's best to increase Tau as well. Below in Figure 10 we see experimental results for changing the parameters on the Paisleys.

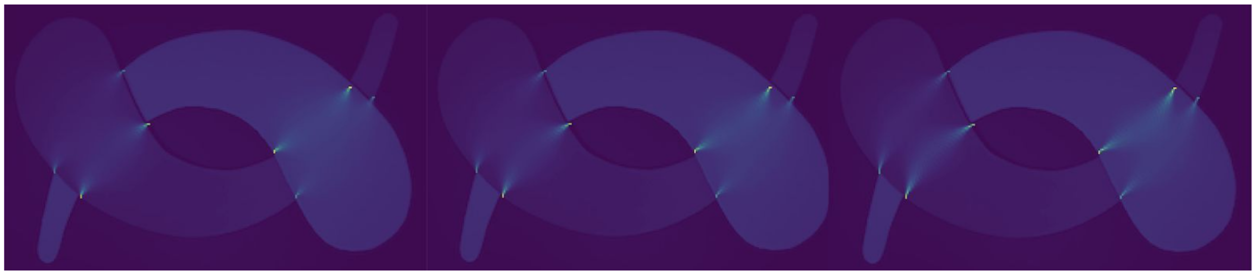

*Figure 10 - Left: Diffusivity = 0.5 Tau = 20* *Middle: Diffusivity = 0.05 Tau = 20* *Right: Diffusivity = 0.005 Tau = 20*

The left image has a diffusivity that is 10x the middle, and the right image has a diffusivity that is 1/10th the middle. As we can see, the left image is more blurry, because the paths are more jagged, and the right image is more rigid. We used a value of Tau=20 here, since the sources and sinks are relatively close. We will explore more experimental results for changing the parameters in the next section.

## 3.4 Underperforming cases:

We will begin this section with a case that does work well, but only under certain circumstances. Consider the Stochastic Completion Field in Figure 11:

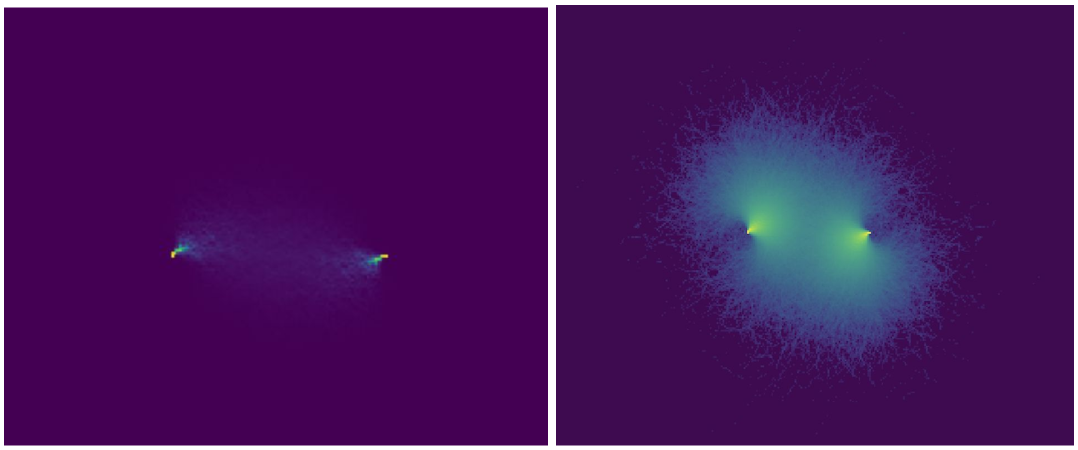

*Figure 11 - SCF with two points distance 80px Left: Diffusivity = 0.05, Tau = 20;*
*Right: Diffusivity = 0.05, Tau = 20*

We have two points of distance 80px from each other, and the Stochastic Completion Field has a diffusivity of 0.05, and a value of Tau = 20. The result seems to work well, and we can see a viable connection between the points. Now, say we wanted to increase the distance between the points to 240px. How would we augment the parameters of Tau and diffusivity to generate a new viable Stochastic Completion Field? One method would be to use a high value of Tau, so that the walks can reach this new distance, and a low diffusivity, because the path between the points is relatively simple. For this, we try a value of diffusivity = 0.005, and Tau = 80. We generate the Stochastic Completion Field in Figure 12:

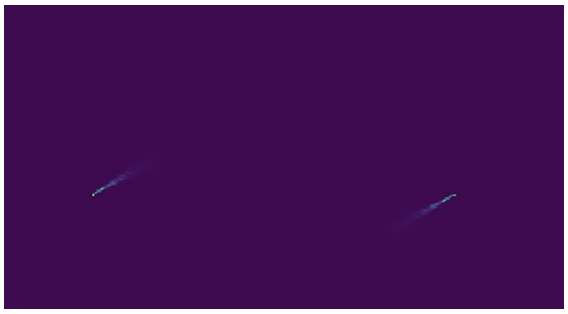

*Figure 12 - SCF with two points distance 240px*
*Diffusivity = 0.005, Tau = 80*

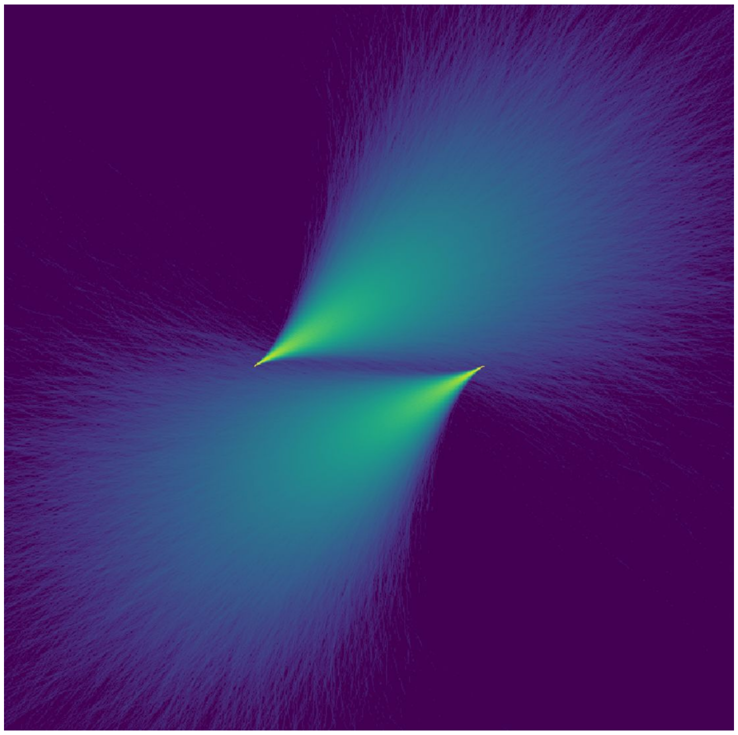

*Figure 13 - Source and sink fields for two points distance 240px*
*Diffusivity = 0.005, Tau = 80*

This field does not look so great, and we can see why in Figure 13. Because the diffusivity is so low, the paths don't get a chance to intersect each other -- even though the value of Tau is big enough that they are able to extend past each other. As a result, the Stochastic Completion Field is weak. We will try another set of parameters. Recall that we only increased the width in the Figure 11 example, so it makes sense that we should only be increasing Tau accordingly. We generate a new Stochastic Completion Field with the same diffusivity as Figure 11 (diffusivity = 0.05), and a value of Tau = 80. We get the result shown in Figure 14.

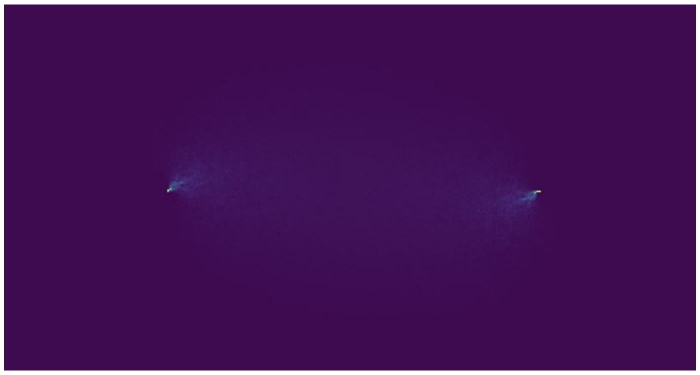

*Figure 14 - SCF with two points distance 240px*
*Diffusivity = 0.05, Tau = 80*

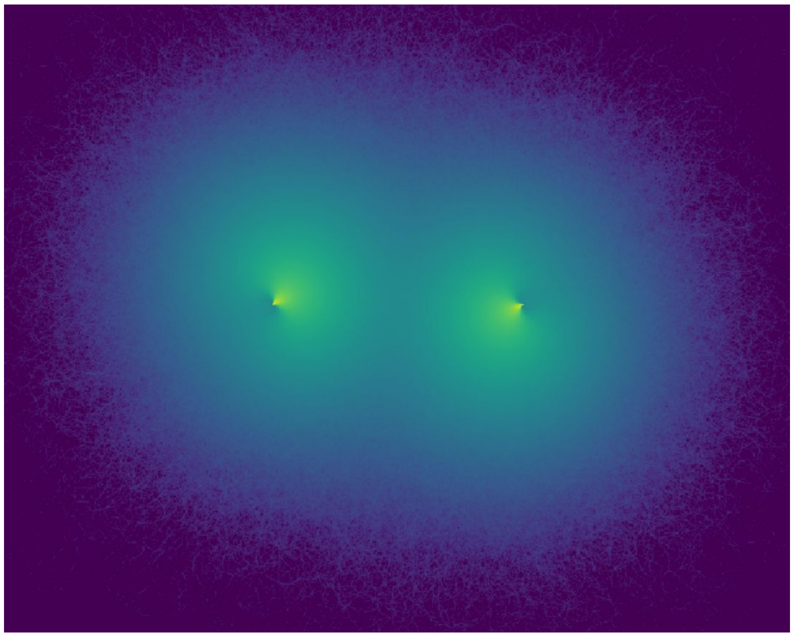

*Figure 15 - Source and sink fields for two points distance 240px*
*Diffusivity = 0.05, Tau = 80*

Figure 14 does not look good either - the image is very blurry and there is no visible contour. When we visualize the source and sink fields for this (Figure 15), we see that they have a lot of variance - much more than the ones in the Figure 11.5 example.

These results make sense when we consider what our Stochastic Completion Field algorithm is doing. At every step, we sample a new angle. So when we take more steps (i.e, increase Tau), we sample more angles, and each new angle sampled makes the path more varied. As a result, there is a relationship between Tau and diffusivity which makes setting hyperparameters difficult. We can say, however, that there exists some underlying relationship between Tao and diffusivity, as path length and path variance are related. Thus, we believe a possible extension to this study could be with the use of machine learning techniques to tweak these hyperparameters.

## 4.1 Future Work

Critically, getting stochastic completion to work on real world images will require a method of retrieving sources' and sinks' locations and orientations. The tools available for detecting keypoints and orientation of edges and features are better now than what Williams and Jacobs had in 1997 [12]. We propose exploring this idea in future work. For an even further extension, we suggest using stochastic completion fields to augment the ability of a deep learning system to recover meaningful contours from an image. Liu presents a strategy for using analytic geometry-based algorithms in conjunction with standard deep learning image translation to create results better than one individual method [?]. A similar approach could be used for combining completion contours on top of a traditional edge detection algorithm with an image translation module that generates contours from natural images.

## 4.2 Conclusion

We revisit the original stochastic completion fields formulation and suggest an implementation with novel contributions to simplify the computational requirements, making the method a viable component of a computer vision pipeline. We show our results on a number of exemplary illusions to demonstrate the completion ability of the algorithm, noting where it fails. By improving on and understanding the behaviour of stochastic completion fields, we take a step in extending this method to make it useful in a real-world computer vision system. Contour completion has valuable use in helping recover shape information that is lost in edge retrieval or by nature of occlusion. Consider a computer vision system that recovers license plate numbers automatically; if a smudge of mud occludes part of a number it may be hard to recognize. Stochastic completion fields could play an important role in preprocessing before passing into mid-level and high-level vision modules in a system that more explicitly considers shape than currently popular deep learning solutions.

## 5. Author's contributions

All Team Members
- General discussion on mathematical theory
- General discussion on algorithm implementation and optimization

Ryan Marten
- Wrote project report Introduction, Background, Methodology, Conclusion
- Conducted literature review
- Visualizations of completion fields and random walks

Sid Gupta
- Ran experiments and analyzed results based hyperparameter changes
- Helper functions that mark sources, sinks, and tangent angles
- Technical writing on analysis of the SCF method

Chandra Gummaluru
- Developed mathematical theory more formally, filling in details
- Implemented algorithm to compute, discretion and rotate the distributions.
- Introduced two novel concepts in the theory to speed up run-time complexity

## References

[1] Williams, Lance R., and David W. Jacobs. "Stochastic completion fields: A neural model of illusory contour shape and salience." Neural computation 9.4 (1997): 837-858. http://www.cs.umd.edu/~djacobs/pubs_files/nc2.pdf

[2] Rezanejad, Morteza, Gabriel Downs, John Wilder, Dirk B. Walther, Allan Jepson, Sven Dickinson, and Kaleem Siddiqi. “Scene Categorization from Contours: Medial Axis Based Salience Measures.” *ArXiv:1811.10524 [Cs]*, November 26, 2018. http://arxiv.org/abs/1811.10524.

[3] Marr, David, and Ellen Hildreth. "Theory of edge detection." Proceedings of the Royal Society of London. Series B. Biological Sciences 207.1167 (1980): 187-217. https://royalsocietypublishing.org/doi/pdf/10.1098/rspb.1980.0020

[4] K. Koffka, Perception: an introduction to the Gestalt-Theorie, Psychological Bulletin 19 (1922) 531. http://library.manipaldubai.com/DL/Perception_an_introduction_to_the_gestalt.pdf

[5] Ullman, Shimon. "Filling-in the gaps: The shape of subjective contours and a model for their generation." Biological Cybernetics 25.1 (1976): 1-6. https://link.springer.com/content/pdf/10.1007/BF00337043.pdf

[6] Parent, Pierre, and Steven W. Zucker. "Trace inference, curvature consistency, and curve detection." IEEE Transactions on pattern analysis and machine intelligence 11.8 (1989): 823-839.

[7] David, Chantal, and Steven W. Zucker. "Potentials, valleys, and dynamic global coverings." International Journal of Computer Vision 5.3 (1990): 219-238. https://link.springer.com/content/pdf/10.1007/BF00126500.pdf

[8] Williams, Lance R., and David W. Jacobs. "Local parallel computation of stochastic completion fields." Neural computation 9.4 (1997): 859-881.

[9] Williams, Lance, et al. "Computing stochastic completion fields in linear-time using a resolution pyramid." Computer Vision and Image Understanding 76.3 (1999): 289-297. http://iar.cs.unm.edu/~williams/pyramid.pdf

[10] Momayyez, Parya, and Kaleem Siddiqi. "3D stochastic completion fields for fiber tractography." 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops. IEEE, 2009. http://iar.cs.unm.edu/~williams/pyramid.pdf

[11] Luong, Tran-Quan "An Exploration of Stochastic Completion Fields" 2003. http://www.cim.mcgill.ca/~siddiqi/COMP-766-2004/projects/quan.pdf

[12] Hu, Shiyan "Stochastic Completion Field with Probabilistic Transition" 2003. http://www.cim.mcgill.ca/~siddiqi/COMP-766-2004/projects/shiyan.pdf

[12] Lowe, David G. "Distinctive image features from scale-invariant keypoints." International journal of computer vision 60.2 (2004): 91-110. http://www.cim.mcgill.ca/~siddiqi/COMP-766-2004/projects/shiyan.pdf

[13] Liu, Difan, et al. "Neural Contours: Learning to Draw Lines from 3D Shapes." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020. https://arxiv.org/abs/2003.10333