The abstracts in this volume comprise the proceedings of the meeting mentioned on the title page. They reflect the authors’ opinions and appear here without change. Their inclusion in this document does not necessarily constitute an endorsement by the editors or the Canadian Human-Computer Communications Society.
Contents

Computer Graphics

Example-Based Print Preview for Laser Cutting
Sarah Kushner, Alec Jacobson

3

PAPARAZZI: Image-driven Surface Editing with Stochastic Multi-View Optimization
Hsueh-Ti Derek Liu, Michael Tao, Alec Jacobson

Fast Winding Numbers for Soups and Clouds
Gavin Barill, Neil Dickson, Ryan Schmidt, David I.W. Levin, Alec Jacobson

Designing Volumetric Truss Structures for Computational Fabrication
Rahul Arora, Alec Jacobson, Timothy R. Langlois, Karan Singh, David I.W. Levin

Theoretical Continuity Improvement of Fairing Algorithms
Xiang Fang, Stephen Mann

Error-Bounded Online Compression of Rigid Body Simulations
Timothy Jeruzalski, John Kanji, Alec Jacobson, David I.W. Levin

Maritime Scene Visualization
Jerry Tessendorf, Liang Gao, Zachary Shore, Colin Reinhardt

Solving Partial Differential Equations on Overlapping Domains
Silvia Sellán, Alec Jacobson

Stipple Removal in Extreme-tone Regions
Rosa Azami, Lars Doyle, David Mould

Augmenting Photographs with Textures Using the Laplacian Pyramid
Lars Doyle, David Mould

Autodef: Non-linear Subspace Simulation for Large Deformation Elastodynamics
Lawson Fulton, Vismay Modi, David Duvenaud, David I.W. Levin, Alec Jacobson

f-Stop: A system for 3D printed stop-motion facial animation
Rinat Abdrashitov, Alec Jacobson, Karan Singh

A Data-driven Approach for Illumination Correction in sRGB Color Space
Mahmoud Afifi, Michael Brown

The Affine Semi-Lagrangian Advection Method
Jade Marcoux-Ouellet, Ryan Goldade, Christopher Batty

A Software Platform for Manipulating the Camera Imaging Pipeline
Hakki Karaimer, Michael Brown

Simplification for Large-Scale Fabrication
Michelle Arkhangorodsky, Yanjun Jiang
Human Computer Interaction

Rigless Skinning for Interactive Vector Animation
Darren Moore, Alec Jacobson, David I.W. Levin

Investigating the Effectiveness of Security-Enhancing Visual Protections for Immersive Collaboration
Stephen Cartwright, Ehud Sharlin, Mario Costa Sousa, Zhangxin Chen

Using Supernumerary Robotic Arms for Background Tasks
Anna Tran, Sowmya Somanath, Ehud Sharlin

Mental Health in Online Communities: How University Students Use Online Communities to Discuss Mental Illness
Denise Y. Geiskkovitch, Andrea Bunt, James E. Young

Conveyor: A Dual-Task Paradigm for Studying VR Dialogue Interfaces
Patrick Dubois, Daniel J. Rea, Kevin Hoang, Meghan Chua, Danielle King, Corey King, James E. Young, Andrea Bunt

Adaptive Visual-Diagnostic Training: User Mental Model Development
Sarah Hoven, Alexis Seniuk, Michel Martel, Hussain Khan, Carrie Demmans Epp, Liam Rourke

Tensai: A Sketch-based Educational Game for Assessing Proper Writing of Japanese Writing Scripts
Dakota Sloniger, Rupen Sakariya, Carlo De Guzman, Jacob Mathews, Erwin Susanto, Jung In Koh, Paul Taele, Tracy Hammond

User-Preferred White Balance of Two-Illuminant Images
Abdelrahman Abdelhamed, Michael Brown

Revisiting Autofocus for Smartphone Cameras
Abdullah Abuolaim, Michael Brown
Example-Based Print Preview for Laser Cutting

Sarah Kushner*
University of Toronto

Alec Jacobson†
University of Toronto

ABSTRACT
Laser cutting is a powerful fabrication tool in which a user can specify vector drawings to be engraved or cut into a material by a laser. This tool is widely used in manufacturing due to its precision and versatility in cutting and engraving small details into a variety of materials. However, the current process of laser cutting from design to fabrication involves guesswork and can end up wasting time and material. We propose a design and visualization tool to create vector drawings that look accurate to what they would be as a laser-cut product. Based on the user’s chosen material and their desired properties of a laser cut, we generate a real-time realistic preview of the finished product as the user edits.

Index Terms: Applied computing—Computer-aided design; Computing methodologies—Image manipulation; Human-centered computing—Visualization systems and tools

1 INTRODUCTION
Laser cutting is becoming a more popular form of fabrication recently. Because there aren’t more standardized tools for it like there are for 3D printing, people must use the laser cutter creatively in order to make what they envision. By working around a laser cutter’s limitations, such as in [5], Mueller et al. bend materials by defocusing the laser and heating them up to make 3D objects. Beyer et al., in [1], make an interactive tool that takes 3D models and separates flat parts of the design into laser-cut plates for fast prototyping. In [4], McCrae et al. slice 3D models into orthogonal planar pieces that can be laser cut and assembled. [3] shows that existing furniture designs can be optimized to use less material while still fitting certain criteria to leave the designs fabricable.

The current process of laser cutting starts with the digital design phase. Programs like Adobe Illustrator and Rhino are used to create vector graphics in such a way that the file can be correctly interpreted by the laser cutter’s driver, such as in Figure 1. The user must draw in very specific colors – only those that come programmed into the driver. These are usually bright, pure colors in RGB format, like red = (255,0,0) for example. After the user is satisfied with their drawing or design, the user sets the stroke width to be extremely thin to the point where the curves are barely visible. This is also a requirement for the driver. But this visual representation, though logical to the driver, is confusing because it does not give any indication to what it will look like in reality.

Next, the user must choose the material they wish to cut or engrave. In our case, this is normally baltic birch plywood. Most laser cutters come with a limited database that holds other types of wood and even different materials like cardboard, paper, and metal. The database has presets for the laser cutter that tells each color to represent a different combination of settings. Alternatively, the user can add to the database and pick their own settings. However, this is not so straightforward. The settings include: power, speed, frequency, and number of passes. These settings are non-intuitive, as the ordinary user may not know what each of them do, much less the relationship between all of them. In our experience, using either the manufacturer-made settings or the user-made settings, it rarely turns out right the first time. Most of the time, adjustments to the settings must be made and the user tries again.

The next element in the design loop is the guessing and checking. This is where the user validates, or more likely, invalidates their settings. The user sends their design to the laser cutter and the result comes out – but usually, it doesn’t end up looking like what they had expected. Common problems include: the laser not cutting all the way through the material when the user wanted it to, unexpected artifacts from the smoke released during a cut, the laser cutter failing to alert the user that something was wrong with their design, etc. In our visualization tool, we imagine functionality that will address all of these problems.

2 METHODS
We propose a system which has the following features to improve the laser cutting process.
2.1 Calibration

The first step which will allow for a print preview to be generated is a calibration of the laser cutter. Every laser cutter is different depending on the brand, the wattage of the laser, the type of laser, the environment in which the laser cutter is used, among many other factors. The material also greatly affects the outcome of the laser-cut product. The same laser cutter with the same settings can cut the same design on two materials, but the results will be distinct. The design will come out differently on metal than it does plastic or leather, for instance.

In order to generalize to a wide range of materials and laser cutters, we use image analogies introduced in [2]. Hertzmann et al. presented a framework for applying a "filter" to images based on an analogy between a set of images.

\[ A : A' :: B : B' \]

If \( A \) is to \( A' \), and \( A \) is a calibration vector graphic and \( A' \) is a photo of this laser-cut piece of material, then given a new vector graphic \( B \), we can create a preview of the design as it would look laser-cut. This will be our \( B' \).

When the user draws a new design, it will become \( B \). Using image analogies, we can generate the \( B' \) which looks like the design laser-cut on wood (or on whichever material the user cut the calibration image). This \( B' \) is updated whenever the user changes the design. This is the basic preview that the user sees.

Our goal in making the calibration design is to cover the largest set of parameters in the fewest amount of cuts and engravings. We need to balance the combinations of settings to represent both a wide range of laser cut appearances and the extreme edge cases. We vary the parameters speed, frequency, and power so that we know how much laser intensity it requires to make marks of different width and color. We can interpolate between to cover the full space. We also make sure the design includes straight lines, curves, and cuts which are placed very close to each other.

2.2 Preview

We take basic features from Illustrator like vector curve drawing, selection, and the fundamental transformations add to the drawing space so that the user can have a more intuitive laser cutting experience. To make this more accessible, we chose to implement it in a web application using JavaScript and its many useful libraries.

To enhance this, we add features which make it a more immersive 3D experience. We generate normal maps of the design and the surrounding wood texture to give the preview depth.

2.3 Explode View

One of the most common mistakes that people make when creating designs to be laser cut comes when the designs are more complication. A design with many pieces often confuses the user and sometimes leads to unwanted pieces missing after they are cut. Sometimes, like stencils, there are “islands” in the design – pieces which are neither connected to the background or the main part of the design. They will fall out after they are cut out of the material. So using connected components, we generate an exploded layer view to show the user.

3 Conclusion

In this extended abstract we proposed an interface to draw and visualize laser cut designs using a variety of techniques. This type of tool does not yet exist for laser cutting and will be a useful improvement to this form of fabrication.

Acknowledgments

The authors wish to thank John Hancock, the system administrator, and Trotec Support.

References

In recent years the availability of consumer virtual reality and fabrication technologies has led to an increased demand for novel 3D geometry. However, the difficulty of creating polished content remains a major roadblock in the supply of this geometry. Although the availability of geometric modeling tools has dramatically lowered the barrier of 3D modeling to the point that even children can create new 3D shapes, these tools do not mitigate the tediousness of polishing a model: introducing fine details, such as geometric texturing, remains a manually exceedingly tedious. Although various techniques exist to apply textures on the surface of existing geometries such as texture mapping and bump mapping, these techniques do not modify the final geometry itself. These traditional rendering techniques are not amenable for fabrication as consumer hardware is focused around reproducing geometric features, and therefore not conducive to truly ubiquitous 3D content creation.

In this paper we develop a novel suite of optimization tools for editing 3D triangle meshes at fine scales driven by their appearances (renderings), which can handle unstructured data. The key ingredient is an efficient method to approximate the derivative of a rendered image of a mesh with respect to the mesh's vertex positions using the GPU. Via the chain rule, this abstracts the design of energy objective functions away from the particulars of geometric representation. This immediately enables minimization of image-domain energies common in image processing (e.g., L0-smoothing) with respect to 3D geometry (see Figure 2).

Figure 1: A smooth model (left) is detailed by transferring geometric details and textures of an image using our image-driven methods (right).

Figure 2: We smooth the shape with L0 gradient regularization. Because this energy encourages an image to be piece-wise constant, we get a piece-wise planar shape.

Figure 3: We transfer geometric details from the input point cloud P to the input shape V through rendering R(P) the point cloud.

Figure 4: Transferring the texture of a surface normal image to a shape.

Figure 5. We transfer geometric textures across surfaces using the image-driven method without providing correspondence information or an example pair.
Moreover, we demonstrate how to cast various geometry processing tasks as image-based optimization problems. By focusing on rendered results we can create textured fabricated geometry even if the reference geometry is not manifold or triangular mesh (see Figure 3). Our generic optimization strategy incorporates Nesterov acceleration and momentum terms to avoid local minima and a stochastic multi-view sampling strategy to ensure full coverage over a shape and without bias.

Our general purpose is to bridge image processing and geometry processing. Under the same regime, we may transfer details to 3D geometry from: 2D renderings, 2D normal images, or 3D geometric details from another model (see Figure 4,5). We demonstrate the power of multi-view optimization for a variety of 3D modeling applications, including: paint-by-numbers style geometric texture synthesis, structure-aware surface hole-filling, appearance-driven mesh refinement, image-based geometric sharpening, and content-sensitive signal transferring.
Fast Winding Numbers for Soups and Clouds

Gavin Barill
University of Toronto
gbarill@cs.toronto.edu

Neil Dickson
SideFX
ndickson@sidefx.com

Ryan Schmidt
Gradientspace
rms@rms80.com

David I.W. Levin
University of Toronto
diwevin@cs.toronto.edu

Alec Jacobson
University of Toronto
jacobson@cs.toronto.edu

Abstract

Inside-outside determination is a basic building block for higher-level geometry processing operations. Generalized winding numbers provide a robust answer for triangle meshes, regardless of defects such as self-intersections, holes or degeneracies. In this paper, we further generalize the winding number to point clouds. Previous methods for evaluating the winding number are slow for completely disconnected surfaces, such as triangle soups or— in the extreme case—point clouds. We propose a tree-based algorithm to reduce the asymptotic complexity of generalized winding number computation, while closely approximating the exact value. Armed with a fast evaluation, we demonstrate the winding number in a variety of new applications: voxelization, signing distances, generating 3D printer paths, defect-tolerant mesh booleans and point set surfaces.

Index Terms: Computing Methodologies—Computer Graphics—Shape Modeling—Point-based models; Computing Methodologies—Computer Graphics—Shape Modeling—Mesh models;

1 Extended Abstract

Determining whether a point is inside or outside of a given shape is one of the most basic geometric questions. Inside-outside segmentation is crucial for: signing distance fields, tetrahedralizing or voxelizing volumes, representing smooth surfaces from point clouds, generating 3D printer path instructions, and surface repair (see figure 1). For analytic shapes and sufficiently clean discrete surface meshes, we can answer the question confidently and quickly. Unfortunately, most surface representations found in the wild are either completely unstructured (e.g., point clouds) or riddled with defects such as open boundaries, duplicated or degenerate geometry, self-intersections and non-manifold combinatorics. The classic winding number determines how many times a planar curve encircles a query point (see, e.g., [3]). Generalized winding numbers [1] extend this concept to oriented triangle meshes suffering from the aforementioned defects. For oriented triangle meshes, this is computed as a sum of signed solid angles $\Omega_t(q)$ of each triangle $t$ subtended at a query point $q$:

$$w_\Sigma(q) = \frac{1}{4\pi} \sum_{t \in \text{Triangles}} \Omega_t(q).$$

For closed watertight meshes, this perfectly reproduces the indicator function (1 inside, 0 outside). For overlapping regions, the winding number measures how many times the region is inside the surface. For holey or non-manifold surfaces, the winding number produces a smoothly varying function revealing a fractional measure of insideness. While simple and robust, a naive implementation of this definition: 1) is slow, requiring $O(nm)$ computation for $n$ queries and an $m$-triangle mesh; and 2) only applies to triangle meshes. For large geometries and interactive applications, inside-outside queries need to be efficient. Existing optimizations for winding number computation either merely use parallelization or make heavy assumptions about mesh connectivity. For large, incoherent triangle “soups” often encountered during scanning or modeling existing methods are too slow. Meanwhile, determining the smooth surface interpolating oriented point clouds is equivalent to extracting the boundary between what the points classify as inside or outside. Most existing point set surface methods are based on grid-dependent discretizations or custom tailored radial basis functions. These methods focus on level-set extraction, but knowing the answer to the inside-outside question has important applications away from the level set (e.g., for signed distances, voxelization, solid 3D printing, etc.).

In this work, we propose a fast method for computing generalized winding numbers on arbitrary triangle soups and point clouds. We begin by deriving a definition of the winding number for oriented point clouds. This directly enables a novel interpolating point set surface representation from the sum of winding number contributions from each point. This sum maintains physical units, avoiding parameter tuning or unnecessary blurring. We then asymptotically improve the performance of this sum with a tree-based algorithm for computing error-controlled approximations of far away points. Analytically integrating our definition for points leads to the familiar generalized winding number for triangles. By applying the same integration to our approximations, we generalize our fast evaluation algorithm to triangle soups as well. We show evidence of the performance and approximation accuracy of our method on a large benchmark, where we achieve up to 1000 speedup for large triangle soups. We test our method in a variety of applications including: point set surfaces, voxelization, mesh cleanup, boolean operations on triangle soups, and signing distance fields. Quickly and robustly answering the inside-outside question allows raw point clouds and triangle soups to travel deeper into the geometry processing pipeline, avoiding lossy representation conversions.

Figure 1: In this paper, we further generalize the winding number to point clouds and propose a hierarchical algorithm for fast evaluation (up to 1000x speedup). This enables efficient answers to inside-outside queries for a wider class of shape representations (top) during a variety of tasks (bottom).
Figure 2: To 3D print the shape represented by the winding field, we convert it to a stack of 2D polygons, which are then filled with toolpaths by the 3D printer software. We extract the polygons using “marching squares” [2] again with a continuation approach. The 3D winding number is used for field evaluations – the 2D winding number along the slice is not the same for open geometry.

As a prototypical example, we show direct 3D printing of point clouds (see figure 2). Our fast evaluation for triangle soups and point clouds will not only improve the performance of all applications already relying on generalized winding numbers, but opens the door to new opportunities. We foresee a rich topic of research exploring how to push raw, unstructured geometric data like soups and clouds farther along the geometry processing pipeline.

REFERENCES
ABSTRACT
We present the first algorithm for designing volumetric Michell Trusses. Our method uses a parametrization approach to generate trusses made of structural elements aligned with the primary direction of an object’s stress field. Such trusses exhibit high strength-to-weight ratio while also being aesthetically pleasing. Unlike traditional approaches to structural optimization, our method produces trusses that can be edited as a post process but retain structural optimality. We also demonstrate the structural robustness of our designs via mechanical testing. Our algorithm permits an exciting combination of control and structural soundness which we believe serves as an important compliment to existing structural optimization tools and as a novel standalone design tool itself.

Index Terms: General and reference—Design; Computing methodologies—Physical simulation

1 INTRODUCTION
A primary objective of engineering is to develop the stiffest possible structure by using the least amount of material. The design of many structures in our everyday life such as bridges, bikes, and buildings such as stadia follow this principle. These structures often form trusses that also manifest an aesthetic appeal, and are therefore of interest to graphics and computational design communities (see [3]).

Automatically designing such minimal structures is challenging as material must be positioned optimally to retain structural soundness. Existing approaches for automatically designing lightweight structures typically formulate the problem as a strength maximization problem, subject to some constraint on the amount of material used in the structure. The two main frameworks for optimal material placement are Topology Optimization [1], which uses a voxel-based representation to explore the design space, and the Ground Structure Method [2] which works on a truss discretization. Both of these methods have inherent limitations, rooted in the requirement of an overprescribed set of design variables (either voxels or bars) as initialization. At a high level, both the strategies involve optimizing for a sparse subset of the initial layout. Both methods are difficult to control, and efficiently editing the output after optimization is still an open problem.

In this paper, we take a different approach to generate aesthetically pleasing, light, and strong structures. Instead of starting with an overprescribed solution and sparsifying it, we use one-dimensional cylindrical structural elements to define a truss, and formulate its design as a fitting problem for these elements. Michell [4] laid the foundations for creating such trusses by proving that for a given material budget, all elements of the optimal (stiffest) truss must follow paths of maximum strain. Structures which fulfill this property are called Michell Trusses. Hence, by aligning the individual elements with the principal stress directions of an object’s stress tensor field, a structurally sound design can be created without needing to fill the entire shape volume with material and later sparsifying it.

We present an algorithm to design Michell trusses inside arbitrary 3D domains. Rather than optimizing an initial guess, we treat truss optimization as a fitting problem, in the vein of recent hex-meshing approaches. Our method requires only a single solve of the static equilibrium equations to compute a continuous stress field. We then use a novel parametrization method to produce a graph of a prescribed resolution where each graph edge is as aligned as possible with the underlying stress tensor field. Our method avoids many of the difficulties of previous methods, its initialization is trivial, and requires no regularization terms to avoid high-frequency artifacts.

Importantly, our approach generates object-spanning material curves that are consistently labelled. This allows us to easily edit our Michell Truss after creation, laying the groundwork for controlable structural optimization. Implementing such customization with existing methods is a grueling task.

2 STRESS-ALIGNED TRUSS NETWORK GENERATION
The input to our method is a tetrahedral mesh, with Dirichlet and Neumann boundary conditions specifying fixed vertices and forces, respectively. We begin by performing standard linear finite element analysis to compute the Cauchy stress tensor field on the mesh. The crux of our algorithm lies in generating a stress-aligned 3D texture parametrization on the tet mesh. Integer isolines of the parametrization can then be extracted as a labelled graph embedded in 3D. Finally, graph elements are inflated to cylinders to get a truss structure.

2.1 Intermediate Frame Fields
Naïvely, a Cauchy Stress tensor field \( \sigma(x) \) can be interpreted as a frame field by representing each tensor by its three eigenvectors.
Because each \( \sigma(x) \) is a Hermitian matrix, its eigenvectors are guaranteed to form an orthogonal basis. However, such a frame field is almost certain to be non-smooth as the direction of each eigenvector can be arbitrarily flipped or interchanged. Our key observation is that the tensor itself is a useful, symmetry agnostic frame field representation and we leverage this notion to fit a frame field.

We define a “good” fit between a frame and a stress tensor as one where the first axis of the frame is aligned with the primary eigenvector of the stress tensor and the other two axes are aligned with the second and third eigenvectors (though it does not matter which aligns with which). Let the stress tensor of the \( j \)-th tet be eigendecomposed as \( \sigma' = Q'(A')^{-1} \). We define the following frame-tensor matching function:

\[
E_{\text{data}}(r_2, r_3) = \|r_2^T S' r_2\|_F + \|r_3^T S' r_3\|_F, \tag{1}
\]

where \( S' = Q'(A')^{1/2} \) and \( r_j \in \mathbb{R}^3 \) is the \( j \)-th direction vector of our frame. This cost function has a set of identical minima at every frame alignment which satisfies our criteria.

Next we need a method for disambiguating the local minima in Equation 1. Typically this is done combinatorially, but here we follow the approach Solomon et al. [5] and instead use a smoothness energy to produce a well-fitted, consistently aligned frame field. While we borrow their Laplacian smoothing term, we avoid their frame field representation as it requires an extra projection step. Instead we represent a frame at the centroid of a tetrahedron using rotation matrices, parameterized via the matrix exponential, \( R = \expm(\sum_j [\omega_j]) \in \mathbb{R}^{3 \times 3} \). Here, \( \omega_j \in \mathbb{R}^3 \) are angular velocity vectors at the vertices of each tetrahedron and the \([\cdot]\) operator cross the product matrix. Combining the data term with Laplacian smoothing gives us the weighted optimization

\[
\omega^* = \arg\min_{\omega} \sum_{i=1}^N E_{\text{data}}(\omega) + \alpha \frac{1}{2}\omega^T L \omega \tag{2}
\]

where \( N \) is the number of tetrahedra in our finite element mesh and \( \alpha \) is a scalar weight. Here, we make an observation that the only purpose of the smoothness energy is to help us choose an appropriate local minima to descend into. Therefore, our final fitting algorithm is Augmented Lagrangian-esque that we repeatedly minimize Eq. 2 with increasingly smaller \( \alpha \) until the cost stops decreasing.

### 2.2 Parametrization Computation

We use our smooth, data-aligned frame field to compute a stress-aligned parametrization from which we will create our Michell Truss. We define \( \Omega \in \mathbb{R}^3 \) as the world space that our object occupies and \( u \in \mathbb{R}^3 \) as a volumetric texture domain. We chose our structural members to lie along the coordinate lines of \( u \) and seek to find a parametrization \( u = \phi(x) : \Omega \rightarrow \mathbb{R}^3 \) that aligns these coordinate lines with our frame field. Formally we seek a \( \phi(x) \) such that

\[
\frac{\partial \phi}{\partial x} r_i = \mathbf{e}_i, \quad \forall i \in \{1, 2, 3\} \tag{3}
\]

at the center of each tetrahedron in our mesh. Here, \( \mathbf{e}_i \) is the column vector representing the \( i \)-th standard basis vector of \( \mathbb{R}^3 \).

This can be restated as a linear system of equations by constructing the discrete directional gradient operator for each tet:

\[
G'(v) = [v_1^T G_1 + v_2^T G_1 + v_3^T G_1], \tag{4}
\]

where \( G_1, G_2 \) and \( G_3 \) are the discrete gradient operators of our tetrahedral mesh, \( v \in \mathbb{R}^3 \) is the direction in which the derivative is to be measured (at the centroid of a tetrahedron) and \( j \) indexes our tetrahedra. We can assemble these local directional derivative operators into global matrices to produce the global operator \( G'(v) \).

We proceed by constructing three directional derivative operators, one for each frame director

\[
G_i = G(r_i) \quad \forall i \in \{1, 2, 3\}. \tag{5}
\]

We frame this problem as a weighted quadratic minimization, with a scalar weight \( \beta \) balancing between two orthogonal notions—uniform spacing of structural members and alignment with coordinate lines:

\[
\phi^* = \arg\min_{\phi} \left\| \begin{bmatrix} G_1 & 0 & 0 \\ G_2 & 0 & 0 \\ G_3 & 0 & 0 \end{bmatrix} \phi - \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \right\|_F^2 + \beta \left\| \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \right\|_F^2
\]

Finally, before extracting the truss by tracing integer isolines of the parametrization, \( \phi^* \) is uniformly scaled to get the desired density.

## 3 Results and Conclusion

We tested our method on a variety of shapes and fabricated the resulting trusses using various manufacturing processes. 2D models were fabricated using a laser cutter, while 3D models were fabricated using a two different additive manufacturing techniques—Fused Deposition Modeling (FDM, Fig. 1c) with soluble support material, and Selective Laser Sintering (SLS, Fig. 1a). Fabricating with such diverse manufacturing processes is tenable using existing approaches. Further, we plan to expand to pipe bending and dowel rod based construction in the future (Fig. 1b).

We performed mechanical tests to experimentally support our theoretical claim that stress-aligned trusses are strong. Our ABS plastic bridge (Fig. 1c), optimized for compression from the top, weighs 140 grams and is able to withstand the weight of an adult human weighing approximately 93 kg (205 lbs). We also utilized laser cutting to build an optimized bike frame for a wooden kids’ bike (Fig. 1d) using 1/4” Baltic Birch plywood. The bike was tested with a 5-yr old weighing 21 kg (46 lbs) and no failures occurred.

**User Control.** Unlike existing approaches for topology optimization and truss optimization, the labelled end-to-end curves produced by our method make our results amenable to user control and modification. Currently, we have implemented density selection—a user can simply select a subset of the labelled curves post-optimization—and vertex snapping, allowing a user to improve visual quality of the results by snapping the integer parameter grid to a set of specified vertices (for example, to sharp corners).

**Conclusion.** We have presented the first algorithm to build volumetric Michell trusses. We evaluated our algorithm by fabricating a variety of structures and performing mechanical tests. We believe that the labelled curves generated by our method also open up avenues for user-controlled structural optimization, and we have demonstrated some initial applications. In the future, we plan to explore a variety of manufacturing methods and structural requirements which can benefit using the user control afforded by our method.

## References


Theoretical Continuity Improvement of Fairing Algorithms

Xiang Fang*  
University of Waterloo

Stephen Mann†  
University of Waterloo

ABSTRACT
We applied some modification to the original fairing algorithms to improve the final surface quality.

1 INTRODUCTION
Scattered data interpolation methods construct surfaces that interpolate locations and first partial derivatives (normals) at the data sites. Often, the data sites are triangulated and spline construction schemes with Bernstein-Bézier triangular patches are used. In general, the minimal degree of Bézier patches required to meet a given order of continuity is high [5]. This degrees can be reduced by triangle split schemes. One of the simplest schemes is the Clough-Tocher interpolant [1], which splits each triangle into three smaller ones. This scheme reduces the minimum degree of \( C^1 \) continuous surfaces from five to three, and has one degree of freedom along each boundary. Kashyap later gave ways to improve the Clough-Tocher interpolants quality by adjusting the available degrees of freedom [4], and in particular by reducing the discontinuity in the crossboundary derivative.

2 FAIRING ALGORITHMS
The Clough-Tocher scheme divides each domain triangle into three. Fig. 1 shows the layout of the control points around a cubic exterior boundary, and Fig. 2 shows the layout of the control points around the cubic interior boundaries inside a domain triangle. Fairing algorithms are local optimization algorithms used to improve the surface quality of Clough-Tocher scheme, which includes exterior fairing across the exterior boundaries (\( P_1P_2 \) in Fig. 1 and \( PQ, QR, RP \) in Fig. 2) and interior fairing across the interior boundaries (\( P_1S, P_1S', P_2S, P_2S' \) in Fig. 1 and \( PS, QS, RS \) in Fig. 2).

Exterior fairing algorithms modify the values of \( l_1 \) and \( r_4 \) in Fig. 1 to minimize the \( C^2 \) discontinuity while keeping the \( C^1 \) continuity across the exterior boundary. Interior fairing algorithms modify the values of all triangular control points and \( S \) in Fig. 2 to achieve \( C^2 \) continuity across the interior boundaries. Kashyap applied both fairing algorithms in turn, repeating several times to obtain a new surface. The first surface in Fig. 3 shows an example of the result of this scheme.

However, if we apply these two algorithms, in turn, an infinite number of times, the surface will switch between two stable states instead of one, i.e., the process does not converge to a single surface. Neither the exterior or interior fairing algorithms construct a \( C^1 \) continuous surface across both interior and exterior boundaries, so one more step of global \( C^1 \) smoothing is required, which will reduce the improvements of the fairing algorithms.

Since the processes of fairing algorithms contain only linear operations, the process of the algorithms can be represented by matrix operations, and the values of all control points can be stored inside a single column matrix. Then the processes of an algorithm can be described as

\[
v \rightarrow Mv + t,\]

where \( M \) and \( t \) are constant. Let \( f^{ex}(v) = M_{ex}v + t_{ex} \) denote the exterior fairing process and \( f^{in}(v) = M_{in}v + t_{in} \) denote the interior fairing process (with the \( C^1 \) smoothing: adjust the value of \( p_5, q_5, r_5 \), and \( S \) to meet \( C^1 \) continuity conditions inside the domain triangles). Then a complete loop can be described as

\[
f(v) = M_{in}(M_{ex}v + t_{ex}) + t_{in}. \tag{1}\]

If this algorithm converges, with any arbitrary initial value, then the values of the control points will eventually reach a limit \( v_1 \):

\[
v_1 = M_{in}(M_{ex}v_1 + t_{ex}) + t_{in}. \tag{2}\]

Writing \( v_2 = f^{ex}(v_1) \), there exist relationships

\[
\begin{align*}
v_1 &= M_{in}v_2 + t_{in} \quad (a) \\
v_2 &= M_{ex}v_1 + t_{ex} \quad (b).
\end{align*}
\]

The surface represented by \( v_1 \) is provided by interior fairing and the surface represented by \( v_2 \) is provided by exterior fairing. In general the limit of \( v_1 \) are not equal to the limit of \( v_2 \), and we need to choose one of them as the final result. However, consider the following proposition:

**Proposition 2.1** Suppose that the repeated application of Equation (1) converges, then Equations (2) hold. In these equations if the subset of the control points updated by (a) in \( v_2 \) and the subset of the control points updated by (b) in \( v_1 \) are mutually exclusive, then \( v_1 = v_2 \).

This means that if each varying control point is updated in only one fairing step, then \( v_1 = v_2 \). It is possible to do some modification to the interior fairing algorithm, such that there is no overlap between two groups of control points in each step, and the final surface will have the continuity properties of both steps. The simplest way is to directly replace the original interior fairing steps by the \( C^1 \) smoothing steps.
Furthermore, there is a quartic version Clough-Tocher scheme, which also provides global $C^1$ continuity surfaces [2]. The quartic exterior fairing algorithm provides $C^2$ continuity across the exterior boundaries. Similar modifications can be applied to this quartic algorithm so that the resulting surface has $C^2$ continuity across the exterior boundaries.

3 Example

Fig. 3 shows the curvature plots of the results provided by the original and modified cubic fairing algorithms, and a quartic surface provided by modified quartic fairing algorithms which has $C^2$ continuous exterior boundaries. The input data sampled from a $4 \times 4$ grid over the domain $[0, 1] \times [0, 1]$ of the Franke’s function No.1 [3]

$$F_1(x, y) = 0.75 \exp\left(-\frac{(9x - 2)^2}{4} - \frac{(9y - 2)^2}{4}\right) + 0.75 \exp\left(-\frac{(9x + 1)^2}{49} - \frac{(9y + 1)^2}{10}\right) + 0.5 \exp\left(-\frac{(9x - 7)^2}{4} - \frac{(9y - 3)^2}{4}\right) - 0.2 \exp\left(-\frac{(9x - 4)^2}{9} - \frac{(9y - 7)^2}{4}\right).$$

Both results are globally $C^1$ continuity, but the second one has lower curvature discontinuity.

References


Figure 3: The difference between the cubic surfaces provided by Kashyap’s (the first) and the modified (the second) fairing algorithms, and a quartic surface provided by modified quartic fairing algorithms (the third).
Error-Bounded Online Compression of Rigid Body Simulations

Timothy Jeruzalski  John Kanji  Alec Jacobson  David I.W. Levin
University of Toronto
tjeruzalski@dgp.toronto.edu

Figure 1: Our physics-inspired compression for rigid body simulations can reduce data size to a fraction of its original frame count. Our method retains all 3D geometry and performs interpolation in time meaning compressed output can be used for a number of tasks such as camera re-positioning, re-lighting and re-timing. Included is an example of a simulation visualized with a realtime WebGL application.

ABSTRACT

Methods to compress simulation data are invaluable as they facilitate efficient transmission along the visual effects pipeline, fast and efficient replay of simulations for visualization and enable storage of scientific data. However, all current approaches to compressing simulation data require access to the entire dynamic simulation, leading to large memory requirements and additional computational burden. In this paper we perform compression of contact-dominated, rigid body simulations in an online, error-bounded fashion. This has the advantage of requiring access to only a narrow window of simulation data at a time while still achieving good agreement with the original simulation. Our approach is simulator agnostic allowing us to compress data from a variety of sources. We demonstrate the efficacy of our algorithm by compressing contact-dominated rigid body simulations from a number of sources, achieving compression rates of up to 360 times over raw data size.

CCS CONCEPTS

• Computing methodologies → Animation; Simulation tools; Physical simulation;

1 INTRODUCTION AND RELATED WORK

Physics simulation has become a bedrock computational tool in engineering and computer graphics. One oft-overlooked facet of physics simulations is that they can produce a huge amount of data [7]. Dynamics simulations are often run at timesteps \( \approx 1 \) millisecond in order to avoid the three I’s: inaccuracy, instability and inter-penetration. Therefore, even short simulations can contain thousands of snapshots of scene state. This inhibits subsequent transmission, browsing and manipulation of the data.

Previous approaches to solving this problem either naively store data at the visual frame rate [1] or require access to the entire simulation sequence to perform compression [2, 4, 5, 9]. Compressing simulation data in an offline fashion, with all data in hand, produces good results but adds a substantial memory and time component to the proceedings. This can be magnified since many of these methods focus on compressing reduced simulation bases which must then be re-simulated in order to generate output [8].

Our solution is to perform compression as the simulation is running, using only a subset of simulation frames at a time. Rather than quantize simulation output to visual frame rate or rely on purely geometric approaches, we interpolate between automatically selected simulation keyframes using physically-inspired functions. We select keyframes to store based on the error they induce over the windowed simulation region.

2 METHODS

Our online compression scheme consists of three components (1) an error metric which quantitatively compares compressed and raw simulation data, (2) a set of physically-based interpolants for rigid body motion and (3) methods to fit these interpolants to incoming simulation data.

As simulation frames are produced, we add them to a sliding window of data, fit our interpolants to this data and then measure the error. If the error exceeds a user-defined threshold we store the end points of the windowed data as keyframes along with the associated parameters of our interpolants.
2.1 Interpolating Functions for Rigid Motion

Inspired by fast methods for space-time optimization [6] we interpolate keyframes using solutions to the governing equations of rigid body dynamics.

\[
\begin{bmatrix}
ml \\
0 \\
\ell _{\text{com}}
\end{bmatrix}
\begin{bmatrix}
a(t) \\
\omega(t)
\end{bmatrix} = \begin{bmatrix}
f_{\text{ext}} \\
\tau_{\text{ext}} = \omega \times \ell _{\text{com}}\omega(t)
\end{bmatrix}, \tag{1}
\]

where \( m \) is the total mass of the body, \( I \) is the 3 x 3 identity matrix, \( \ell _{\text{com}} \) is the constant, body space inertia matrix, \( a \) is the linear, \( \omega \) is the angular acceleration and \( f_{\text{ext}} \) and \( \tau_{\text{ext}} \) are a constant external force and torque respectively.

2.2 Function Fitting

Next we describe the procedures by which we fit our interpolants to our simulation data. We solve the following optimization problem.

\[
\min_{\tilde{q}, \tilde{\omega}} \frac{1}{2} \int_{t_0}^{t_1} \| q_{\text{com}}(t) - \tilde{q}(t) \|^2 dt \
\text{s.t. } q_{\text{com}}(0) = \tilde{q}(0), \quad \tilde{q}_{\text{com}}(1) = q_{\text{com}}(1)
\tag{2}
\]

Where \( q_{\text{com}}(t) \) are the states from the simulation and \( \tilde{q}(t) \) is the fit.

**Translation.** We use the analytical solution for ballistic motion to interpolate CoM position which is of the form \( q(t) = a_0 t^2 + v_0 t + p_0 \).

Here \( a_0, v_0 \) and \( p_0 \) are the 3D acceleration, initial velocity and initial position of the body over a time interval beginning at \( t_0 \). Our translation interpolant can be efficiently fit to our windowed simulation data by solving a linear least squares problem for each rigid body in the scene.

**Rotation.** For the rotation fitting high order polynomial fits were explored, but the best compression performance came from the linear interpolation. A slight modification to simple SLerp is able to encode multiple rotations around an axis with a single axis by also storing angular velocity. Using the starting and ending rotations for the object, with the additional angular velocity component to describe how many times to rotate.

2.3 Error Metric

We measure center-of-mass (CoM) and rotational error separately using sum of squared distance between CoM position and orientation respectively, integrated over the current simulation window from \( t_0 \) to \( t_1 \). We continue extending \( t_1 \) until \( E_{\text{com}} > \epsilon_{\text{com}} \) or \( E_{\text{rot}} > \epsilon_{\text{rot}} \) where \( \epsilon_{\text{com}} \) and \( \epsilon_{\text{rot}} \) are set manually. We augment this with a running average peak detector on the second derivative of the errors to place keyframes at sudden events such as collisions.

3 RESULTS

Our method achieves higher compression rates than simply storing data at low frame rates. We accomplish this while still retaining essentially infinite temporal precision allowing for interesting editing operations such as adding slow motion to movie scenes. We also demonstrate that our method works well for a number of popular simulation packages (Houdini [10], SCSim [11] and Bullet [3]) achieving between 360x and 5x compression rates. Again, our method is simulator agnostic so these results were obtained with no changes to our algorithm or implementation. Finally, we show that our compression times are negligible compared to the required simulation time (our slowest example takes only 10% of the simulation time). Because our streaming compression is pipelined with the simulation, the actual waiting time imposed on the user would be much less in practice.

We also performed visual comparison with standard algorithms which are often applied for animation compression. Our comparisons to Principal Component Analysis (PCA) and Douglas Peucker line fitting make it clear that our compressed output is of much higher visual fidelity and more closely tracks the input simulation. Both previous methods exhibit distracting artifacts with the rotational errors exhibited by PCA being particularly egregious. More impressive is that we achieve these superior results without needing to analyze the entire set of simulation data (unlike the methods we compare against).

REFERENCES


<table>
<thead>
<tr>
<th>Example</th>
<th># Bodies</th>
<th>Simulation Time</th>
<th>Compressing Time</th>
<th>Uncompressed Size @ 30fps</th>
<th>Uncompressed Size @ 240fps</th>
<th>Uncompressed Size @ 1000fps</th>
<th>Compressed Size @ 30fps</th>
<th>Compressed Size @ 240fps</th>
<th>Compressed Size @ 1000fps</th>
<th>Compression Ratio</th>
<th>Collision Frames</th>
<th>Error Bound Frames</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drop</td>
<td>1</td>
<td>5.42s (realtime)</td>
<td>0.17s</td>
<td>63 KB</td>
<td>20 KB</td>
<td>2 KB</td>
<td>528 B</td>
<td>168x</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Bounce</td>
<td>1</td>
<td>5.11s (realtime)</td>
<td>0.39s</td>
<td>357 KB</td>
<td>86 KB</td>
<td>6.6 KB</td>
<td>2.3 KB</td>
<td>155x</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Throw</td>
<td>1</td>
<td>4.28s (realtime)</td>
<td>0.50s</td>
<td>230 KB</td>
<td>60 KB</td>
<td>6 KB</td>
<td>5.9 KB</td>
<td>45x</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Spin</td>
<td>1</td>
<td>3.53s (realtime)</td>
<td>0.08s</td>
<td>50 KB</td>
<td>5 KB</td>
<td>416 B</td>
<td>126x</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Tomahawk</td>
<td>1</td>
<td>4.56s (realtime)</td>
<td>0.46s</td>
<td>63 KB</td>
<td>63 KB</td>
<td>240 B</td>
<td>12 KB</td>
<td>28x</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Gyroscopic Spin</td>
<td>1</td>
<td>25.3s (realtime)</td>
<td>0.17s</td>
<td>317 KB</td>
<td>94 KB</td>
<td>12 KB</td>
<td>28x</td>
<td>121</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Falling Tower</td>
<td>107</td>
<td>155s</td>
<td>5.5s</td>
<td>88 MB</td>
<td>21 MB</td>
<td>21 MB</td>
<td>244 KB</td>
<td>366x</td>
<td>1578</td>
<td>3473</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>SCSim Bunnies</td>
<td>200</td>
<td>30 mins</td>
<td>0.6s</td>
<td>43 MB</td>
<td>402 KB</td>
<td>71x</td>
<td>6754</td>
<td>4550</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Houdini example</td>
<td>55</td>
<td>10s</td>
<td>0.36s</td>
<td>530 KB</td>
<td>104 KB</td>
<td>53x</td>
<td>1222</td>
<td>641</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Large City Scene</td>
<td>6619</td>
<td>110 mins</td>
<td>1.63s</td>
<td>21 MB</td>
<td>147x</td>
<td>260542</td>
<td>134860</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 1: Results of compressing a number of rigid body simulations with our method. We report both total compressed size and time to compress, along with the number of key frames dropped due to error-bound exceptions and due to peak detection.

Finally we compare to storing the simulated data at truncated frame rates.
ABSTRACT

Index Terms: Applied Computing—Earth and Atmospheric Sciences; Applied Computing—Arts and Humanities; Computing Methodologies—Model Development and Analysis; Computing Methodologies—Simulation Theory; Computing Methodologies—Physical Simulation;

1 INTRODUCTION

Assembly, simulation, and rendering of maritime scenes is of use in both entertainment and engineering applications. The feature film industry heavily employs CG oceans and water simulations in a broad range of films, such as Titanic, Poseidon Adventure, 300, Life of Pi, Pacific Rim, and many more. The quality and detail of these scenes, driven by creative storytelling priorities, is so high that audiences willingly suspend disbelief. An important reason for the high quality of this work is that the process uses scientifically established techniques in fluid simulation that are frequently altered for storytelling purposes. The output of the fluid simulations are subjected to post-simulation modification to satisfy the storytelling goals for the visual appearance of the result.

For engineering applications, suspension of disbelief is inadequate as a criterion of success. Creating a maritime scene for engineering applications requires that the individual components are grounded in scientific principles and processes. Assembly of scenes from these components must follow scientific principles as well, while preserving the validity of each component. Ultimately, validation against previously established results and/or experimental data is very important.

However, if a maritime visualization is to be highly accurate scientifically, it should also be visually believable under appropriate conditions. The entertainment industry, while shunning concern about science, has significant experience in how to practically assemble and manage complex simulations and run those simulations efficiently to arrive at visually believable outcomes.

A software system has been under development for the purpose of simulating and visualizing maritime scenes for engineering applications, while also employing appropriate software, techniques, and experience from entertainment applications. The project, named Gilligan [3], integrates multiple simulation and rendering software components that originated in visual effects applications. The integration framework is a collection of Python modules that promote scientifically valid steps, and allow growth and modification of the feature set. The features required of Gilligan exceed what is available in entertainment software, so other components have purely engineering and scientific origins.

2 GILLIGAN FRAMEWORK

The concept of an environmental scene simulator/visualizer is similar to a multiphysics simulation engine. An environment is conceptually organized in terms of many components representing the natural phenomena of interest. For maritime scenes, major components of interest include the ocean surface, atmosphere, clouds, sun and other possible lights, water volume, and many other potential components. These components must be integrated in a scene description framework that controls the simulation process for the components that have simulation aspects, and controls the rendering process for the production of images of the scene. Table 1 lists a collection of features and components that a maritime scene simulation/visualization system like Gilligan should have.

In Gilligan’s design, the scene description is a relatively small collection of Python classes and modules that represent every component and control operation. Components that involve significant
computational resources invoke C++ and/or CUDA-based shared objects with Python bindings to the scene description. For example, the ocean surface component is both a C++ library for FFT ocean surface waves [1] and a python API for its use. Additionally, interactive waves [2] are a separate C++ library with a very similar python API. An additional Python class provides and API for merging two or more layers of ocean surface and interactive waves into a Python object with the same API as the underlying components.

Volumetric clouds are represented as fields of density and optical properties, assembled from a mix of modeling and simulation algorithms that give natural, fractal-like structure [4]. Clouds can also appear in a rendered scene that uses a HDRI background.

Integration into graphics applications is feasible and intended via the Python API. For example, Python scripts could be used to integrate Gilligan into Maya, providing users with a familiar user interface in Maya, but physically accurate behavior in Gilligan.

3 INITIAL DEMONSTRATIONS

Gilligan is an ongoing development. The features that have been implemented are italicized in Table 1. In this current implementation, Gilligan has a global illumination path tracing renderer. Materials are attached to objects in the scene, and a material is a collection of one or more C++ shaders that manage optical properties such as BRDFs, GI statistics and path continuation, and volumetric shading. Ray-geometry intersections are efficiently handled via BVH.

Fig. 1 illustrates several of the current features in Gilligan. A cargo ship near the horizon is obscured in a hydrostatic haze layer. The sky is an HDRI photo, and underwater light scattering is also a hydrostatic haze and upwelling color. The camera bobs three feet above the waves and the renderer uses path tracing global illumination. The ocean surface is assembled from three layers: (1) a swell surface with waves on scales greater than 4 meters; (2) a wind wave layer with waves less than 30 meters, and (3) an interactive layer that propagates wave disturbances from the breaching vehicle. These layers are broken down in Fig. 2.

Fig. 3 is a scene with the camera at a 20 meter altitude above the surface. As the vehicle transits through waves, whitecaps form due to the interactive surface disturbance. There are whitecaps throughout the environment, and a three dimensional cloud constructed from pyroclastic displacements of the shape of the vehicle. The path trace global illumination insures that cloud radiance fully interacts with the ocean surface.

**ACKNOWLEDGMENTS**

This work was supported by SPAWAR Systems Center Pacific Contract N66001-16-P-6865.

**REFERENCES**

**ABSTRACT**

Partial Differential Equations (PDEs) arise in many areas of science, from physics to biology to computer graphics. In the latter, they can be used for data smoothing [3], shape deformation [2] and even computing geodesic distances [1], among many others. By far the most common PDE present in computer graphics and geometry processing applications is the Poisson equation ($\Delta u = h$ for some known function $h$) or its particular case, the Laplace equation ($\Delta u = 0$). Our objective is usually to find the solution for these equations in a two-dimensional or three-dimensional domain $\Omega$ subjected to some Dirichlet boundary condition $u|_{\partial \Omega} = g$ for a known function $g$.

The preferable method to numerically approximate solutions to these equations consists in discretizing the domain in a mesh of small elements (triangles in 2D, tetrahedra in 3D) in a way that the analytical equation can be converted into a quadratic energy minimization problem where the variables are the values of the function $u$ at the elements’ vertices, which we denote by the vector $u$. This problem can be solved with the standard techniques of quadratic programming to obtain the minimizer $u^*$, from which we can build a piece-wise linear function which can be shown to be a discrete approximation of the PDE’s analytical solution $u$.

The accurateness of this approximation is directly related with the goodness of the discretization, measured mainly by its coarseness (size of the elements) and its regularity (aspect ratio of edges of each element). In practice, manually constructing quality meshes for complex three-dimensional domains is an arduous and generally undesirable process, while the automatic software available fail too often.

Most of the times, the final complex domain $\Omega$ we intend to solve the PDE on (see ‘pawn’ example on the right) can be described via the union of very simple primitive domains $\Omega_i$, the discretizations of which are either known or easy to find. Inspired by this, we present a method for solving PDEs on the final domain which makes use only of the meshes of the primitive shapes, thus eliminating the need for discretizing the complex domain itself.

Our first step is constructing a combined quadratic energy which functions as an analogous of the single-domain energy described above. To achieve this, we add the individual energies we would obtain from solving the PDE on each primitive and weigh them in a way that we avoid accounting for the same area twice in the zones where the primitives intersect.

Simply minimizing this combined energy subject to the original PDE’s boundary condition is not enough. The problem lies precisely in those regions where the primitives intersect, where we would obtain various different solution values $u^1 \ldots u^n$ (corresponding to the $n$ intersecting primitives) which need not be consistent with each other. We will explore different methods of imposing this consistency requirement within the energy minimization in a way that the individual minimizers on each primitive do approximate the analytical solution on the final domain.

For clarity, let us restrict ourselves to the case where $n = 2$, i.e. our final domain can be written as $\Omega = \Omega_1 \cup \Omega_2$ and we wish to formulate a constraint that guarantees us that our energy minimizers $u_1$ and $u_2$ take similar values in the region $\Omega_1 \cap \Omega_2$. Since the vertices of each domain’s discretization need not coincide, enforcing an equality between $u_1$ and $u_2$ is not as straight-forward as it may seem. In order to be able to make this comparison, we will need to construct the piece-wise linear functions $u^1(x)$ and $u^2(x)$ from the vertex values $u_1$ and $u_2$.

Enforcing $u^1|_{\Omega_1 \cap \Omega_2} = u^2|_{\Omega_1 \cap \Omega_2}$ (equality on all points in the intersection) can easily be shown to be too harsh a constraint since it will effectively restrict our minimization to the space of functions which

* e-mail: sgsellan@gmail.com
† e-mail: jacobson@cs.toronto.edu
are linear over the intersection (see Fig. 3). This is an artificial restriction which persists even as the discretizations are refined. We will refer to this type of undesirable consequences as locking.

The first formulation of this consistency that we present is that which enforces equality between both functions only at all of the discretizations’ vertices. This requirement can easily be expressed as a linear equality constraint to add on to our previous quadratic program. We show that this method also results in locking when attempting to solve the Poisson equation, although it performs surprisingly well when solving the Laplace equation (see Fig. 2). The reason for this lies in the similarity between our imposed linear constraint and the energy associated with the Laplace equation (usually known as the cotangent laplacian or discrete laplacian), which we intuitively show by a study of each’s eigenmodes.

After examining the failure of the above’s method for the Poisson equation, one may suggest softening the requirement by exchanging the linear equality constraint for a quadratic penalty term to add to the PDE’s energy we are minimizing. Naturally, the severity of the penalty will have to be governed by a positive parameter $\lambda$. If $\lambda$ is small, $u_1$ and $u_2$ become increasingly disconnected; on the other hand, if $\lambda$ is too large, adding this term is equivalent to the linear equality constraint seen to fail above. While we show that there exist values of the parameter for which this method accurately solves a Poisson equation, the optimal $\lambda$ depends on the equation itself and the characteristics of the discretization.

Finally, we present our boundary method, which consists in enforcing equality between the solutions only at the vertices in the intersection’s boundary, which can also be expressed as a linear equality constraint to add to our quadratic program. We show that, contrary to the previously suggested formulations, this requirement results in a convergent solver which is parameterless and consistently achieves low error.

We also compare this method’s convergence to the one we would obtain if we used a standard solver on a discretization of $\Omega$ (blue line in Fig. 2). We find that the expected loss in accuracy is small compared to the benefit of not requiring a mesh of the final domain, thus validating our method.

We end by showing a sample of computer graphics applications of our solver for complex three-dimensional domains constructed as the union of simpler ones, and present possible generalizations of our method to other PDEs and to domains generated via Constructive Solid Geometry, which includes other basic set operations apart from unions, such as intersections and subtractions of basic shapes.

REFERENCES


Stipple Removal in Extreme-tone Regions

Rosa Azami* Lars Doyle† David Mould‡
Carleton University

1 INTRODUCTION

Stippling uses small dots or circles to represent the image content. Stippling and other illustrative visualization techniques such as halftoning, hatching and silhouette have been widely used in artistic and scientific applications.

In computer graphics and NPR, stippling techniques have focused on optimizing the point distribution and the point size [2] or tone preservation [1]. Although algorithmic stipple placement tried to maintain some characteristics such as regularity and density of the stipples, yet various problems are not solved for stippling.

Stippling methods require enormous numbers of stipples to represent black areas of an image. An image with extensive dark regions requires vast quantities of stipples and consumes excessive memory in storing them. We plan to cover each region with a polygon instead of expending huge quantities of stipples. Another problem is the placement of scattered stipples in lighter regions; such stipples are not expressing any structure or tone of the image. Such redundant stipples are distracting, and we identify these stipples and remove them.

We applied our method on different photographs with different compositions and lighting conditions. Results are shown on the second page of this abstract. Our two main contributions are as follows:

• Illustrating dark regions: We identify the dark regions of the image and cover them with solid polygons instead of stipples. To eliminate discontinuities, we set smooth boundary conditions and reconstruct the image in a small band surrounding each dark region.

• Controlling stipple placement in light regions: We identify and remove the redundant stipples on light regions.

2 METHOD

This work addresses two main issues. First, traditional stippling methods match the tone of dark areas by filling them with huge numbers of stipples; we propose instead covering each dark area with a polygon. Second, tone-based stippling can produce isolated stipples in light regions; such stipples are distracting and may not represent any specific image content. We propose a postprocessing method that identifies isolated stipples and removes them from the image. We show our algorithm in figure 1.

For a given image, first we identify the dark regions that would require many stipples. We determine these regions by thresholding the low intensities of the image. We plan to represent these regions by polygons rather than filling them with numerous stipples. Similarly, we threshold the image’s high intensities to white so as to avoid the necessity of putting stipples there; in such regions, stipples will be infrequent and may appear spurious to a viewer. Figure 1(a) shows an input image, and (b) shows the dark regions obtained by thresholding.

At cutoff thresholds, the transitions between the black (or white) regions and the nearby areas are discontinuous. In order to obtain smooth transitions, we reconstruct the nearby regions by solving a Poisson equation [4]. When the reconstructed image is passed to the stippling algorithm, the thresholded pixels are labeled as “done”, thus preventing stipples from being placed there. Figure 1(c) illustrates the reconstructed gray image and (d) the result of applying stippling to the reconstructed image.

Once the stippling is complete, we remove additional potentially spurious stipples appearing in the bright regions of the image. Such stipples usually do not contribute much information. We remove isolated stipples through two post-processing steps. In the first step, we remove sufficiently isolated stipples: those that have at most one other stipple nearby. In the second step, we remove additional stipples deemed not to contribute to the image content. We compare two quantities: pixel intensity of the target stipple, and the average intensity of the original image in a range surrounding the stipple location. If the difference between the quantities is small, we discard the stipple; if it is large we keep the stipple. If the difference between the quantities lies in an intermediate range, we do further analysis, basing our decision on the number of stipples nearby: when there are at least four stipples within a given distance, the stipple is considered to be part of the image structure and hence kept, whereas if there are fewer, the stipple is removed.

In the bottom row of Figure 1, (f) shows the final result and a close-up of the original, polygon filling and stipple removal steps, respectively.

3 RESULTS

Figure 2 illustrates results from our method. All results use the structure-aware stippling method of Li and Mould [3]. Our reconstruction of the boundaries near dark regions not only increased the contrast but also provides a smooth transition in placement of stipples from dark regions to their surroundings. We also had a significant reduction in size of the vector graphics file of images with large dark regions. For example, the SVG file of the old man declined from 12MB to 4MB. Moreover, we reduced the number of stipples significantly by covering the dark regions with solid black polygons. Removing the isolated stipples also reduced the number of stipples; it also affected the image contrast and improved viewer perception of the image by eliminating distracting stipples.

REFERENCES

Figure 1: Illustration of the steps of our method. a) original image, b) threshold mask, c) reconstructed image from Poisson blending, d) the results from filling polygons, e) our final result after removing isolated stipples and f) close-up of original stippling, polygon filling and stipple removal.

Figure 2: Our results. From upper left: Face, Bird, Old man and their corresponding original stippling in bottom row.
Augmenting Photographs with Textures Using the Laplacian Pyramid

Lars Doyle*  David Mould†
Carleton University Carleton University

1 INTRODUCTION
Texture plays an important role in our appreciation and understanding of images. It can serve as a visual replacement for the tactile qualities that images lack, and prompt the photographed subject matter to appear more lively and interesting. Many photographers have realized this potential and have used textures to enhance their images. Often the intent is to transform an ordinary photograph into an artwork that contains characteristics of paintings, e.g., cracks, background materials, and brushstrokes. Other times, the intent is to create a “vintage” look, brought on by artificially weathering digital images.

Image blending is the obvious way to combine a texture and an image. However, there are disadvantages: contrast is reduced, colors may be altered, and edges fade. In addition, it is often desirable for new textures to follow the orientation of the structures in the original image. For example, performing texture augmentation on a photo of a pet, an appealing option is to align the example texture with the fur orientation. Accomplishing this manually would be tedious and likely beyond the capabilities of many users.

In our approach, we preserve the edges and textures from the original image. In fact, the result of our method will exhibit the large-scale characteristics of the original, yet it will be stylized with high-frequency details that are taken from an external texture source. Our method provides a novel tool that digital artists can use to create images that deliver a rich visual experience to the viewer.

2 CONTRIBUTION
We have two major contributions:

1. We propose merging an image with a texture by mixing the coefficients of their Laplacian pyramid representations. The coefficient mixing uses the smooth maximum function [3]. Combining an image and a texture in this way retains the structural characteristics from the image while augmenting it with the fine-scale details of the texture.

2. We propose an irregular tiling based on SLIC super-pixels [1] for patch-based texture synthesis. In the context of combining an image and a texture, the SLIC patches not only provide an irregular structure, making artifacts less noticeable, but also align with image edges, further concealing defects in the synthesized texture when it is integrated with the image.

3 METHOD OVERVIEW
Our texture augmentation system requires two inputs: an input image and a texture image. The texture image is produced by means of a patch-based texture synthesis system, which uses SLIC super-pixels as the atomic texture units. In addition, we design our method to preserve the local orientation of the input image in the output texture.

We use the image structure tensor [2] to obtain orientation information, which we then use to rotate texture patches into alignment with the input image.

Next, having synthesized a texture image, we merge it with the original input image. Our method builds Laplacian pyramids for both the input image and the texture image, then computes a result pyramid that combines coefficients from both inputs. Our coefficient mixing strategy uses the smooth maximum function:

\[
SM(u, v, k) = \ln\left(\exp\left(ku\right) + \exp(kv) - 1\right)\frac{1}{k},
\]

which returns a value larger that the larger of \(u\) and \(v\). Here, \(u\) and \(v\) represent Laplacian coefficients from the input image and the synthesized texture, respectively. The parameter \(k\) controls the degree of amplification. The intuition behind this approach suggests that where the input image contains salient information, such as edges and textures, these should be preserved in the output. Conversely, where the auxiliary texture is relatively more prominent, this it should be carried over to the output image.

4 RESULTS
Figures 1 and 2 show two results from our texture augmentation method. In Figure 1, we can notice that the outlines of the building remain crisp against the newly added texture, which is most visible in the low-frequency areas of the sky. In Figure 2, we combine two auxiliary textures into the input image using a continuous mask as a guide. In this example, we can see how the added texture is oriented parallel to coherent features, such as the model’s jawline and jewelry.

Note: An article [4] based on this work has been accepted for publication in the journal The Visual Computer.

REFERENCES
Figure 1: Result from our texture augmentation method. Left: original image; right: our result, showing the texture example in the inset.

Figure 2: Result using two auxiliary textures. From left to right: spatial mask and two texture examples are shown from top to bottom, original image, our result, detail of result.
Non-linear Subspace Simulation for Large Deformation Elastodynamics

Lawson Fulton†
University of Toronto

Vismay Modi‡
University of Toronto

David Duvenaud§
University of Toronto

David I.W. Levin¶
University of Toronto

Alec Jacobson†
University of Toronto

ABSTRACT
High-resolution deformable body simulations are a staple of offline visual effects, but see less use in interactive settings. Full scale simulations are too slow and previous reduced models either do not capture rich details of elastic deformation or are unstable at run-time. Furthermore, the highest quality reduced models rely on tedious rig construction or careful training. In our paper, we propose using a non-linear reduced space based on a fully connected autoencoder neural network. In contrast to linear models, a very small latent space using our reduction can better capture detailed elastic simulations. We complement our reduced space with a reduced energy integration resulting in a simulator that can run in real time for large models. We demonstrate the performance, stability and ease-of-use of our method for a variety of deformation scenarios.

Index Terms: General and reference—Design; Computing methodologies—Physical simulation; Dimensionality reduction and manifold learning;

1 INTRODUCTION AND RELATED WORK
Physics-based animation is a transformative tool for computer graphics. The goal of physics-based animation is to create artists to create compelling, realistic animations with less effort, allowing the laws of physics to generate complex motions that would be tedious or impossible to animate. The power of this idea is illustrated by the fact that almost every blockbuster movie now features characters, cities and liquids that are physically simulated.

It is also telling that highly detailed, large scale simulated characters and deformable objects are rarely seen outside of visual effects for movies or television (i.e., domains that tolerate long runtimes required for visually plausibility). For applications which require interactive runtimes, compromises are made and shortcuts are taken, usually, dropping the fidelity of the simulation. While physical models that can generate plausible animations exist, they are currently too slow to simulate. Over the last decades, the work on accelerating solid mechanics simulations for animation has made leapfrog advances. These works take two broad forms, solver-based approaches (which attempt to improve the linear algebraic operations at the heart of a physics simulator) or discretization-based approaches (which try to reduce the degrees-of-freedom in a computational mesh). One of the most successful and beloved acceleration techniques to speed up elastodynamics is linear subspace simulation, a discretization-based technique which performs simulations in a small reduced space constructed from either the linear vibration modes of a deformable object [6] or PCA basis of simulation snapshots [5].

However, linear modal analysis comes with one major issue: accurate representations of motions that span highly non-linear regions of

\[ \frac{\partial}{\partial t} U^T M U z = U^T f_{int}(U z) + U^T f_{ext} \]  

(1)

By reducing the size of the system to integrate, we can often gain a substantial performance benefit.

2 METHODS
Reduced space simulation improves performance by representing an object’s configuration \( q \) as a function of a smaller number of variables \( q = U z \), where \( U \in \mathbb{R}^{n \times r} \) forms the basis of a reduced space of dimension \( r << n \). By projecting into the reduced space we can express the equations of motion as

\[ L_0(q) = \| D_q (E_\theta(q)) - q \|^2 \]

where \( D_q : \mathbb{R}^r \rightarrow \mathbb{R}^n \) is the decoder network, and \( E_\theta : \mathbb{R}^n \rightarrow \mathbb{R}^r \) is the encoder network and \( \theta \) are the network parameters.

Unfortunately, training this network directly on the simulation data yields unacceptable visual results, fraught with high-frequency
4. **Time Integration**

In order to solve this problem using our reduced space, we reformulate the equations of motion as an optimization at the position level using the substitution \( \mathbf{q} = \tilde{f}(\mathbf{q}_1 - \mathbf{q}_0) \) in conjunction with the fact that \( \mathbf{q} = UD(z) \) which yields

\[
\begin{align*}
\delta(\mathbf{z}) &= D(\mathbf{z}) - 2D(\mathbf{z}_0) + D(\mathbf{z}_{-1}) \\
\mathbf{z}_1 &= \arg\min_z \frac{1}{2} \delta(\mathbf{z})^T M \delta(\mathbf{z}) + h^2 \Psi(UD(z)) - h^2 D(z)^T \mathbf{f}_{\text{ext}}
\end{align*}
\]

where we can define the reduced mass matrix \( \tilde{M} = U^T M U \), reduced internal force vector \( \mathbf{f}_{\text{int}} = U^T f_{\text{int}} \) and reduced external force vector \( \mathbf{f}_{\text{ext}} = U^T f_{\text{ext}} \). We can minimize Equation 5 efficiently using the L-BFGS algorithm. In our case we warm start each iteration with \( \tilde{H} = J_D(\mathbf{z}_0)^T K J_D(\mathbf{z}_0) \) where \( K = U^T K U \) and \( K = \frac{\partial^2 \Psi(\mathbf{0})}{\partial \mathbf{q}^2} \) is the reduced stiffness matrix at the objects rest state and \( J_D(\mathbf{z}_0) \) is the Jacobian of \( D(z) \) evaluated at the beginning of the current time step. \( \tilde{H} \) is fast to compute because it only relies on our autoencoder Jacobian and the rest state stiffness matrix.

5. **Reduced Space Integration**

Even with the L-BFGS solver above, our simulation algorithm still has one major bottleneck, the computation of the reduced internal forces, \( \mathbf{f}_{\text{int}} \). Naively computing these forces requires evaluating \( \tilde{f} \) and then assembling across all the elements in our simulation mesh. The most popular way to avoid this problem is via optimized cubature [1, 7] in which an optimal subset of mesh elements are selected, along with corresponding scalar weights, such that the weighted sum of their force evaluations yields a force estimate with low error.

These methods attempt to estimate a non-negative set of cubature weights, in keeping with standard cubature rules from numerical integration.

To estimate the complete energy vector we build a linear reduced space, \( S \), by performing PCA on the set of stacked energy vectors \( \Psi \) from each simulation data pose. Now, given a subset of mesh elements \( T \), we stack their energies into the vector \( \Psi \) and search for the low dimensional point \( \mathbf{a}^* \) such that \( \Psi = S \mathbf{a}^* \) best matches this data. This allows us to express the total energy in the object as

\[
\Psi(q) = \sum \Psi(q)^s = s^T \mathbf{a}^* = \ldots s^T (S^T S)^{-1} S^T \Psi(q) = \mathbf{w}^T \Psi(q),
\]

The challenge then becomes finding a suitable subset of mesh elements \( T \) which minimizes the error of the PCA completed energy \( \Psi^* \) over the training set in the least squares sense:

\[
T = \arg\min_{\tilde{T}} \frac{1}{2} \| \Psi_T - \Psi^* \|^2_2,
\]

This is a best subset selection problem which we solve using an off-the-shelf simulated annealing implementation. Our reduced integration scheme completes our fast, non-linear reduced simulation.

6. **Results and Conclusion**

Table 1 contains performance comparisons between our method and traditional linear reduced space simulation. Please see our poster and live demo for more details.

We have presented the first neural network-based, non-linear reduced space simulation of large elastic deformations. Our algorithm leverages a new type of autoencoder for learning deformations, an L-BFGS-based dynamics solver and an original optimized cubature approach that exhibits excellent results for dynamic, large deformation objects. Our method offers comparable (sometimes better) performance to standard, linear subspace methods but offers dramatically better robustness during interactive simulation.

**References**


f-Stop: a system for 3D printed stop-motion facial animation

Rinat Abdrashitov*  
University of Toronto

Alec Jacobson†  
University of Toronto

Karan Singh‡  
University of Toronto

Figure 1: f-Stop workflow: input computer animations of a deformable 3D object (a); are segmented and deformed into parts that can be seamlessly joined (b); Each part is independently used to compute a replacement set that is representative of the deforming part in the input animations (c); the replacement set is iteratively optimized and a mapping from each frame of the input animations to a member of the replacement set is computed (d); The replacement library is processed for object assembly and 3D printed for use in stop-motion animation (e)

ABSTRACT

We present the first system for optimizing facial animation sequences to produce a set of replacement faces for use in 3D printed stop-motion animation. The input to our system is any sequence of topology invariant deforming meshes (typically faces). Inspired by the workflow adopted by recent stop-motion films we first segment the mesh along curves of minimal deformation. We present a novel algorithm to zero out the deformations along part boundaries, so that 3D printed replacement sets for each part can be interchangeably and seamlessly assembled together. For each part we independently optimize an energy function to find a replacement set using a graph-cut technique. The size of the replacement sets can be automatically computed to adhere to a printing budget, quantitative deviation from the original animation, or user controlled for a qualitative approximation. We further instrument the replacement sets with plugs, so that parts can be efficiently swapped. Our approach has the potential reduce the printing time and cost by more than 100× for stop-motion animated shorts or feature films.

Index Terms: Computer Graphics—Animation—Mesh Models;

1 INTRODUCTION

Stop-motion is a traditional animation technique that moves a physical object in small increments between photographed frames, to produce the illusion of fluid motion. As with animation in general, arguably the most expressive part of a character is its face. Replacement parts in general have been the standard approach to the stop-motion animation of expressive deformable objects, in particular faces. With the advent of 3D printing, replacement animation has become a bridge between the disparate worlds of digital computer animation and physical stop-motion, and is increasingly used as the preferred technique for producing high-quality facial animation in stop motion film [4].

Faces and 3D models in general are created digitally (or physically sculpted and scanned) to produce a library of replacements that cover the expression range of the 3D model. This library, typically containing thousands of variations of a deformable model are then 3D printed and cataloged. Additional post-processing maybe required, including sanding down edges, smoothing inconsistencies, and hand painting the 3D prints. The replacement library is then ready to be used in stop-motion sequences [1]. The cost in terms of printing and post-processing time, material, storage and money are prohibitive. Each character of Laika’s stop-motion feature film Coraline could have as many as 15,000 faces and up to 250,000 facial expressions.

2 METHOD

Given a deformable 3D object $O$, we aim to produce a 3D printed replacement library, that captures the expressive deformable range

*  
e-mail: rinat@dgp.toronto.edu

†  
e-mail: alec@cs.toronto.edu

‡  
e-mail: karan@dgp.toronto.edu

Figure 2: Modern stop-motion films such as Laika’s ParaNorman amass libraries of thousands of 3D printed replacement faces.
of $O$. This can be provided by way of a number of input animations $A$ involving $O$ (that will be emulated using stop-motion), or a representative sampling of the deformation space of $O$. In either case we can concatenate all input (animation clips and discrete samples), into a single input sequence $X = [x_1, x_2, \ldots, x_n]$ of $n$ frames. Each $x_i$ is a topology invariant mesh, represented either by its vertex positions or blendshape weights (rig parameters) that define its deformed shape. For a model represented by $m$ parameters thus $X = [x_1, x_2, \ldots, x_m] \in \mathbb{R}^{m \times n}$.

### 2.1 Object Segmentation

Many deformable objects like faces have localized regions of deformation separable by near rigid boundaries. Stop-motion facial animation often segments a head into an upper and lower face just below the eye-line, and a rigid back of the head. As there are aesthetic and assembly considerations to segmenting a deformable object $O$ (represented as a triangle mesh), we let the user specify the number of segments $s$, and seed vertices for each of the $s$ segments.

The output is an assignment of each triangle of the mesh to a part in $\{p_1, \ldots, p_s\}$ and a minimal deformation of each mesh/blendshape in $X$, so that the boundaries between segments of objects in $X$ are rigid (with zero spatial gradient). The deformation ensures that replacement parts can be assembled seamlessly (with first order continuity).

We optimize segment boundaries by minimizing an energy that penalizes cutting along edges that move a lot in $X$. Let $\bar{v}$ represent the average position of each vertex $v$ of the mesh $O$ across the $n$ frames of $X$. We minimize an energy that punishes cutting along often moving edges:

$$b(i, j) = \begin{cases} c_{ij} & \text{if faces } i \text{ and } j \text{ share an edge and part } p_i \neq p_j \\ 0 & \text{otherwise} \end{cases}$$

$$c_{ij} = \sum f \in J \frac{\| \tilde{e}_{ij} \| (1 + 1)}{\| \tilde{e}_{ij} \|}$$

where $\tilde{e}_{ij}$ is the length of the edge between faces $i$ and $j$ at frame $f$, $\tilde{v}_j$ is the vector from the $j$th vertex at frame $f$ to that vertex’s average position.

Once faces are assigned to parts, we need to force the vertices along the seam to be constant throughout the deformation and have zero gradient. We solve for a smooth displacement per frame that pulls the seam values to a constant value. We constrain every vertex $v$ along part boundaries in $O$ in all frames of $X$, to their average value $\bar{v}$ (i.e., the values that minimize average displacement across all frames). We then solve for the displacement that minimizes the integrated squared spatial Laplacian of changes over the shape, subject to seam value and derivative constraints. In the continuous setting, this would be measured as:

$$\arg \min_{\tilde{q}} \int_{\Omega} (\Delta (q - \bar{v}))^2 \, dA$$

subject to: $q(x) = \bar{v}(x)$ along the seam and $V q(x) = V\bar{v}(x)$ along the seam

on a triangle mesh, and this is effectively implemented using the mixed finite element method described by Jacobson et al. [2]. This deformation from Equation 2, preserves surface detail while producing a smooth, slowly varying change from the part boundaries.

### 2.2 Replacement library and mapping

As the replacement library for each of the $s$ object parts is computed independently, we now assume a single object part, for brevity of notation. Given $X \in \mathbb{R}^{m \times n}$, we aim to output a replacement dictionary $D$ that is a set of $\ell$ instances of the object $O$, $D = \{d_1, d_2, \ldots, d_\ell\} \in \mathbb{R}^{m \times \ell}$ and a sequence of $n$ labels assigning each input $X_i, i \in 1, \ldots, n$ to a corresponding replacement in the dictionary $\ell = \{\ell_1, \ell_2, \ldots, \ell_n\} \in \{1, \ldots, \ell\}^n$. We optimize for $D$ and $\ell$ that best approximate the input geometry and the change in input (e.g., the discrete velocity) between consecutive frames for inputs that come from animation clips (recall the animation indicator cut(i) for $i \in 1, \ldots, n$):

$$\min_{D, \ell} \frac{1}{2} \| X - DS \|_F^2 + \frac{\lambda}{2} \| XG - DSG \|_F^2,$$

where $S \in \{0, 1\}^{\ell \times n}$, $S_{ij} := \begin{cases} 1 & \text{if } \ell_j = i, \\ 0 & \text{otherwise} \end{cases}$

where $\lambda$ balances between shape accuracy and velocity accuracy, and $G \in \mathbb{R}^{\ell \times (n-1)}$ is a sparse matrix computing the temporal forward finite difference of the $m$ shape parameters.

Since fixing the labels $\ell$ also fixes the replacement matrix $S$, finding the optimal dictionary amounts to minimizing a quadratic least squares energy. The optimal dictionary $D$ is a solution to a large, sparse, linear system of equations.

Fixing the dictionary $D$ and optimizing for the labels $\ell$ is less trivial. By the definition of the temporal finite derivative matrix $G$ we may rewrite the objective function in Equation (3) as a sum of unary terms involving the independent effect of each label $\ell_j$ and binary terms involving the effect of pairs of labels $\ell_j$ and $\ell_k$ corresponding to the $j$th and $k$th animation frames:

$$\frac{1}{2} \| X - DS \|_F^2 + \frac{\lambda}{2} \| XG - DSG \|_F^2 = \sum_{j=1}^n u_j + \sum_{j=1}^n \sum_{k=1}^n b_{jk},$$

where $u_j := \frac{1}{2} \| (x_i - d_\ell_i) \|_F^2$ if $|j - k| = 1$, otherwise.

The binary term $b_{jk}$ satisfies the regularity requirement described by Kolmogorov and Zabin [3]. Problems of this form are efficiently solved using graphcut-based multilabel optimization (e.g., alpha-expansion).

### 3 Conclusion

Stop-motion animation is a traditional art-form that has seen a surge of popularity with the advent of 3D printing. f-Stop is the first attempt at an end-to-end solution to the research problems in the creation of stop-motion animations using computer animation and 3D printing. We hope this paper will stimulate new research on the many problems encountered in the area of stop-motion animation.

### References


A Data-driven Approach for White-Balance Correction in the sRGB Color Space

Mahmoud Afifi
Department of Electrical Engineering and Computer Science, Lassonde School of Engineering, York University, Toronto

ABSTRACT

Once a photo is taken, it is nearly impossible to correct the color cast caused by using the wrong white-balance setting. This is because the white-balance procedure happens as an early processing step on the camera (applied to the sensors raw image) and is followed by a number of non-linear photofinishing manipulations to generate the final standard RGB (sRGB) image. The non-linear photofinishing steps in the camera pipeline make it difficult to correct the white-balance in an sRGB image with existing techniques. In this preliminary work, we present a data-driven approach to address the sRGB illumination correction problem. We rely on thousands of pairs of incorrectly white-balance sRGB images and their corresponding correct image. We use this dataset of image pairs (incorrect and correct white-balance images) to generate a color correction mapping in the sRGB color space for a given input image. To the best of our knowledge, this is the first work addresses the problem of illuminant correction in the sRGB color space, outperforming all commercial solutions for this problem.

Keywords: White balance, illumination correction, sRGB

1 INTRODUCTION

The human visual system has the ability to correct the color cast caused by the dominating scene illuminant. This explains in part why an apple appears red to under sunlight, incandescent light, and fluorescent light – even though these illuminations are significantly different in their spectral profile. Camera sensors, however, do not have this ability and as a result, computational color constancy is required to be applied on-board the camera. This procedure is typically termed "white-balance". There is a large body of work addressing computational constancy in raw space [1]. White-balance is applied early in the camera processing chain and is critical, not only for aesthetic reasons, but also for vision-based tasks, such as image retrieval, object tracking, skin detection, and image forensics [2-5].

One overlooked issue with white-balance is that it is applied early in the processing chain on what is termed the raw image. Cameras have a number of processing steps that convert the raw sensor response to the final output image. These collective steps result in a final output that is saved in a standard RGB (sRGB) color space [6]. Most of the steps applied after white-balance are nonlinear in nature. If the initial white-balance is computed incorrectly on the camera, it is hard to undo this afterwards in the sRGB image due to the non-linear photofinishing operations applied on the camera (see Fig 1). In fact, no existing white-balance algorithm even attempts to address correction in sRGB images. This is a significant problem that results in huge numbers of photographs being discarded due to the wrong white-balance.

Our work proposes a data-driven approach to correct sRGB images that have been rendered using the wrong white-balance setting. As part of this effort, we have generated a dataset of over 60,000 images from different cameras that have been rendered into sRGB with the camera's possible white-balance settings, including incorrect and correct settings, as well as different color rendering modes. Given an improperly white-balance sRGB image, we find incorrectly white-balanced training images with similar color distributions to the input image. Based on our training data, we construct a nonlinear color correction that is applied to the input sRGB image. This straightforward strategy gives good results that significantly outperforms existing solutions that attempt to apply post rendering white-balance.

2 METHODOLOGY

Given an sRGB input image that has undesirable color casts rendered with an unknown white-balance setting through unknown nonlinear color manipulations, the goal is to obtain an output image that looks similar to the same image that has had its white-balance correctly applied. Using our dataset of thousands of pairs of incorrectly white-balance sRGB images and their corresponding correct image, we compute a mapping function that maps the colors of the incorrect white-balanced image.

Our method is a data-driven approach in the way that we use a projected version of the color distribution of the input image to find the nearest training color distributions (NTCD). From the NTCD, we generate a new color correction map based on weighting the NTCD’s mapping functions. We found that, the 9 × 3 polynomial color correction (PCC) [8] is effective and has an acceptable amount of error. By applying the generated PCC map to the input sRGB image, we could effectively correct the illumination casts.

3 RESULTS

Our method outperforms the “exact” white balance solution which refers to the diagonal white balance correction using a white reference point from the scene as shown in Fig. 2. We used the mean square error (MSE) and the mean angular error (MAE) to evaluate our method; where both MSE and MAE of our method less than the errors of the exact white balance solution and the commercial software packages’ (e.g., Adobe Photoshop) solutions, as shown in Table 1.

It is worth noting that the exact white balance solution is the best case can achieved using the current illuminant estimation algorithms.
We have proposed a data-driven method to correct sRGB images that have been incorrectly white balanced. The approach relies on a massive amount of images rendered with all possible white balance settings and picture styles found on several different makes and models of cameras. Given an input sRGB image with the wrong white balance, we retrieve images with similar color distributions in the dataset. From these retrieved images, a color correction is computed to correct the input image. Quantitative and qualitative experiments demonstrated the effectiveness of this approach over existing solutions to this problem.

4 Conclusion

We have proposed a data-driven method to correct sRGB images that have been incorrectly white balanced. The approach relies on a massive amount of images rendered with all possible white balance settings and picture styles found on several different makes and models of cameras. Given an input sRGB image with the wrong white balance, we retrieve images with similar color distributions in the dataset. From these retrieved images, a color correction is computed to correct the input image. Quantitative and qualitative experiments demonstrated the effectiveness of this approach over existing solutions to this problem.

References


<table>
<thead>
<tr>
<th>Method</th>
<th>MSE</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Exact&quot; white-balance</td>
<td>867.56</td>
<td>8.71</td>
</tr>
<tr>
<td>Adobe Photoshop</td>
<td>328.68</td>
<td>6.38</td>
</tr>
<tr>
<td>Ours</td>
<td>280.64</td>
<td>5.87</td>
</tr>
</tbody>
</table>

Table 1. The mean of mean square error (MSE) and mean angular error (MAE) of the "exact" white balance solution (the best results could be obtained using the traditional solutions), the Adobe Photoshop's auto-color correction function, and our results. The lower is better.

Figure 1. (A) An sRGB image with the wrong white balance applied. There are two achromatic regions highlighted in red and yellow. (B) and (C) show standard white balance correction applied to the sRGB image using different reference white points. (D) and (E) show the Bradford transform-based correction as described in Eq. S4 using the same reference white points. (G) Ground truth sRGB image with the correct white balance applied.

Figure 2. The ground truth image in (B) is captured in (A) by a DSLR camera with a wrong white balance setting and a different camera picture style. The result of the Bradford transform-based white balance correction using the exact white reference point is shown in (C) and our result is shown in (D).
The Affine Semi-Lagrangian Advection Method

Jade Marcoux-Ouellet a, Ryan Goldade b, Christopher Batty b

* e-mail: jade.marcoux-ouellet@uwaterloo.ca
† e-mail: rgoldade@uwaterloo.ca
‡ e-mail: christopher.batty@uwaterloo.ca

David R. Cheriton School of Computer Science
University of Waterloo

ABSTRACT

We propose a new Eulerian advection method for regular grid fluid simulation called ASLAM, for Affine Semi-Lagrangian Advection Method. Our method is less dissipative than the traditional semi-Lagrangian advection. It adapts ideas from the affine particle-in-cell method (APIC) to the particle-free Eulerian setting.

Index Terms: Computing methodologies—Computer graphics—Animation—Physical simulation;

1 INTRODUCTION

Lagrangian particles are often preferred in fluid animation for their ease of advection: transport of properties by particles is readily understood and simple to compute with numerical integration techniques. However, purely Lagrangian frameworks are less well-suited than their Eulerian counterparts to enforcing a divergence-free velocity field, a crucial factor in simulating incompressible flows. On the other hand, Eulerian methods, like classic semi-Lagrangian advection [3], suffer from excessive numerical dissipation. Hybrid methods, such as “FLIP” [4], have been developed to exploit the advantages of both frameworks by performing advection on particles and pressure projection on a grid, and using transfer operations to map quantities from particles to grid nodes and vice versa.

A recent variant of such hybrid schemes, dubbed APIC [2] for Affine Particle-In-Cell, exhibits both good stability and reduced dissipation. It conserves both linear and angular momentum while preserving an affine velocity field by augmenting particles with gradient data. In the current work, we apply similar ideas to a fully Eulerian simulation using a set of temporary fictitious particles, which are back-traced through the vector field, as in the traditional semi-Lagrangian advection scheme introduced by Starn [3]. However, our particles carry both field value and gradient information, similar to APIC. This approach exploits the same dissipation-reducing behaviour of APIC, compared to basic semi-Lagrangian, while also dispensing with the need for a persistent particle set. Our method is motivated in part by recent work by Ferstl et al. [1] that observed that a significant expense of hybrid methods is associated with particles, which justified their decision to revert to cheaper Eulerian approaches on the liquid interior.

2 THE AFFINE SEMI-LAGRANGIAN ADVECTION METHOD

We closely follow the formulation of Jiang et al. [2] (Section 6) for the transfers between particles and grid, repurposing their component-wise vector formulation for the scalar case we currently consider. To adapt it to our Eulerian particle-free setting, we instead use fictitious particles and combine the transfer operations with the semi-Lagrangian back-tracing operation itself, as we describe below.

Table 1: Notation used in this document. Preference is given to the scalar form of the equations, where the advected quantity is a scalar $q$. For the advection of vector data, $q$ can refer to each component of the advected vector quantity.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q^i_0$</td>
<td>Quantity at grid node $i$ at $t = n\Delta t$</td>
</tr>
<tr>
<td>$q^{bp}_{bp}$</td>
<td>Quantity at the traced-back stencil particle $p$ at $t = n\Delta t$</td>
</tr>
<tr>
<td>$q^{n+1}_i$</td>
<td>Updated quantity at grid node $i$ at $t = (n + 1)\Delta t$</td>
</tr>
<tr>
<td>$m_i$</td>
<td>Mass at grid node $i$</td>
</tr>
<tr>
<td>$m_p$</td>
<td>Mass of stencil particle $p$</td>
</tr>
<tr>
<td>$w^{n+1}_{p \rightarrow p}$</td>
<td>Weights from grid node $i$ to stencil particle $p$ at $t = n\Delta t$</td>
</tr>
<tr>
<td>$w_{p \rightarrow i}$</td>
<td>Weights from stencil particle $p$ to grid node $i$</td>
</tr>
<tr>
<td>$x_i$</td>
<td>Position of grid node $i$</td>
</tr>
<tr>
<td>$x_p$</td>
<td>Position of stencil particle $p$ before back-tracing</td>
</tr>
<tr>
<td>$x_{bp}$</td>
<td>Position of stencil particle $p$ after back-tracing</td>
</tr>
<tr>
<td>$S_i$</td>
<td>Set of particles included in the stencil of grid node $i$</td>
</tr>
<tr>
<td>$\nabla p$</td>
<td>Gradient of $q$ at the traced-back stencil particle $p$ at $t = n\Delta t$</td>
</tr>
</tbody>
</table>

Figure 1: Example of a stencil for a cell-centered grid. Full circles • represent grid nodes positions while empty circles ◦ correspond to stencil particles positions. In this example, each grid node $i$ is associated with eight stencil particles, whose positions are fixed. The stencil $S_i$ has thus $|S_i| = 9$ as we include the grid node $i$ itself. Note that stencil particles can be shared between grid nodes.

We will use subscripts $p$ and $i$ to denote quantities stored on particles and grid nodes, respectively. The subscript $bp$ indicates the traced-back position of a stencil particle $p$. The grid nodes’ positions depend on the type of grid data to be used (i.e., vertex-, cell-, or face-centered). The superscript $n$ is reserved to indicate which time step the value of the superscripted quantity is taken from. Vectors are lower-case and bold. Table 1 summarizes all of our notation.

Assuming for the moment that we have a set of local fictitious particles along with data already assigned, we can update the quantity $q$ at grid node $i$ according to the following transfer operation from particles to grid nodes

$$q^{n+1}_i = \frac{1}{m_i} \sum_{p \in S_i} m_p \sum_{p \rightarrow i} \left( q^{n}_{bp} + \left( \nabla p \right)^T (x_i - x_p) \right)$$

(1)

referring to Table 1 for definition of symbols. In essence, this is a weighted average of Taylor expansions from the particle data to the node in question. The “masses” $m_i$ are computed as

$$m_i = \sum_{p \in S_i} m_p w_{p \rightarrow i}.$$  

(2)
We apply our advection method on two different scenarios in which

we adopt the stencil illustrated in Fig. 1. While the weights

and their gradients at the traced-back positions

are computed based on interpolation weights

and their gradients at the traced-back positions

in much the same manner as APIC, using

\[
q_{bp}^n = \sum_i q_i^n w_{i\rightarrow bp}^n, \quad (3)
\]

\[
e_{bp}^n = \sum_i q_i^n \nabla w_{i\rightarrow bp}^n. \quad (4)
\]

We use \( m_p = 1 \) for simplicity, since we focus on scalar data, and we adopt the stencil illustrated in Fig. 1. While the weights \( w_{i\rightarrow bp}^n \)

are given by the bilinear interpolation kernel, the weights \( w_{bp\rightarrow i} \)

are uniformly distributed between all \( p \in S_i \). We use the (explicit)

midpoint method for all back-tracing operations.

3 Results

We apply our advection method on two different scenarios in which

an initial torus-shaped scalar density field is advected by simple

fixed velocity fields. In Fig. 2 we compare the results against basic

semi-Lagrangian advection and the analytical solution. In the first

scenario, we use a steady translational flow with velocity \( v = (0, 1) \).

In the second scenario, we use a steady rigid rotational flow around

the center \( c \) of the grid with \( v = (c_y - y, x - c_x) \). We observe visually

that our method is less dissipative than semi-Lagrangian advection:
notice the reduced blurring and better preservation of the empty

region at the center of the torus.

In Fig. 3, the maximum \( L_\infty \) error is plotted against time. While

the error inevitably increases with time, our method fares better

than semi-Lagrangian advection. However, as shown in Table 2,

our method is currently an order of magnitude slower than the semi-

Lagrangian method. This is to be expected, since we back-trace

more points (i.e., the additional stencil particle positions) and we

have to compute the gradient vector \( e_{bp}^n \) for all back-traced points.

We anticipate that denser or larger stencils would be less efficient,

but could also improve accuracy. While we take advantage of ficti-

tious particles being shared between neighbouring nodes to avoid

recomputing the same values, quantities such as \( q_{bp}^n \) and \( e_{bp}^n \) could

potentially be computed in parallel to further improve performance.

4 Conclusion

We proposed an advection method that hybridizes APIC with the tra-

titional semi-Lagrangian method to reduce dissipation while avoid-

ing the need for a persistent particle set. Similarly to APIC, an

affine description of the advected quantity is accounted for during

the advection. In the future, we plan to apply our method to the

time-varying velocity vector field itself to yield improved Eulerian

simulation of smoke and other fluid phenomena.

The original semi-Lagrangian method prevented “blow up” by

bounding the updated quantity based on the simple interpolant used.

Further investigation of the stability of our method is needed, since it

is not obvious that these guarantees still hold. We would also like to

explore the effects of different stencil choices (varying radii, particle

count, particle position), compare against other common Eulerian

advection schemes, and explore extensions to higher order.

Acknowledgments

This work was supported in part by the Natural Sciences and En-

gineering Research Council of Canada (NSERC), the Fonds de

recherche du Québec —Nature et technologies (FRQNT) and a

David R. Cheriton Graduate Scholarship.

References


2015. doi: 10.1145/2766996


121–128. ACM Press/Addison-Wesley Publishing Co., New York, NY,


A Software Platform for Manipulating the Camera Imaging Pipeline

Hakki Can Karaimer
York University, Toronto

ABSTRACT
There are a number of processing steps applied onboard a digital camera that collectively make up the camera imaging pipeline. Unfortunately, the imaging pipeline is typically embedded in a camera’s hardware making it difficult for researchers working on individual components to do so within the proper context of the full pipeline. This not only hinders research, it makes evaluating the effects from modifying an individual pipeline component on the final camera output challenging, if not impossible. This paper, which appeared in 2016 in the European Conference on Computer Vision [1], presents a new software platform that allows easy access to each stage of the camera imaging pipeline. The platform allows modification of the parameters for individual components as well as the ability to access and manipulate the intermediate images as they pass through different stages. We will detail our platform design and demonstrate its usefulness on a number of examples.

Keywords: Camera processing pipeline, computational photography, color processing.

1 INTRODUCTION

Digital cameras are the cornerstone for virtually all computer vision applications as they provide the input to our algorithms. While camera images are often modeled as simple light-measuring devices that directly convert incoming radiance to numerical values, the reality is that there are a number of processing routines onboard digital cameras that are applied to obtain the final RGB output. These processing steps are generally performed in sequence and collectively make up the camera imaging pipeline. Examples of these processing steps include Bayer pattern demosaicing, white-balance, color space mapping, noise reduction, tone-mapping and color manipulation. Many of these processing steps are well-known research topics in their own right, e.g. white-balance, color space mapping (colorimetry), and noise reduction.

Although cameras are the most prominent hardware tools in computer vision, it is surprisingly difficult to get access to the underlying imaging pipeline. This is because these routines are embedded in the camera’s hardware and may involve proprietary image manipulation that is unique to individual camera manufacturers. This is a significant drawback to the research community. In particular, it forces many researchers to work on topics outside the proper context of the full imaging pipeline. For example, much of the work targeting white-balance and color constancy is performed directly on the camera-specific raw images without the ability to demonstrate how it would affect the final output on the camera. Another example includes noise reduction (NR) targeting sensor noise. On a camera, NR is applied before many of the non-linear photo-finishing routines (e.g. tone-curve manipulation), however, researchers are generally forced to apply NR on the final non-linear sRGB image due to a lack of access to the camera pipeline. This presents a significant mismatch between assumptions made in the academic literature and real industry practice.

Contribution: We present a software platform to allow easy access to each stage of the imaging pipeline. Our approach operates on images saved in DNG raw image format which represents the unprocessed sensor response from the camera and the starting point for the camera processing pipeline. Our platform allows images to be opened and run through a software rendering API that parallels the onboard processing steps, including the individual processing components and their associated parameters. Specifically, our platform provides API calls that allow the modification of processing components’ parameters and full access to the intermediate images at each processing stage. Such intermediate images can be modified and inserted back into the pipeline to see the effect on the final output. The proposed software platform can be integrated with other software such as Matlab and provides a much needed environment for improving camera imaging, or performing experiments within the proper context of the full camera imaging pipeline.

2 PLATFORM OVERVIEW

Our platform uses images that are saved in the Adobe Digital Negative (DNG) format. While this format is not yet supported by many of the DSLR cameras, it is currently being supported by the newer Android phones that implement the Camera 2 API. With Android’s adoption of DNG, the number of raw images captured by mobile devices are expected to increase significantly. The DNG image format not only contains the raw image data but also contains meta-data that specifies parameters (e.g. scalar values or a 1D or 3D look up table (LUT)) intended to be used by different stages in the processing pipeline.

Our platform is made possible by rewriting the interface of the open source Adobe DNG SDK software [2] that provides a full software implementation of a camera pipeline to convert the DNG raw image to its final sRGB output. While this is an engineering feat, the implementation is non-trivial. The standalone Adobe DNG SDK is not designed to allow changes to the parameters of the individual stages, instead the SDK uses the values in the DNG files meta-data directly. Thus the processing pipeline had to be decomposed into its individual stages and API calls designed to access and modify the underlying parameters.

Figure 1-(A) overviews the processing steps that are available in the proposed camera imaging platform. The top shows the steps with the associated parameters used by each of the components while Figure 1-(B) shows the intermediate images at each stage in the pipeline. In the following, we detail each stage and its associated parameters that can be modified. The type of parameters used by the individual stages are also discussed. In the case of a 1D LUT, the same LUT is applied to each color channel individually.

Stage 1: Reading the raw image (Params: None) The unmodified raw image is read from the DNG image file. This is the unprocessed image produced by the sensor that is still in its mosaiced Bayer pattern format.
Stage 2: Black light subtraction and linearization (Params: Level values or 1D LUT) The unmodified raw image is linearized such that its values range from [0-1] in the processing pipeline. Many cameras provide a BlackLevel parameter that represents the black level of the sensor that deviates from 0 due to sensor noise. This is often image specific and related to other camera settings, including ISO, gain, etc. An additional WhiteLevel (maximum value) can also be specified. If nothing is provided, the min and max value of all intensities in the image is used to normalize the image. Another alternative is to provide a 1D LUT to perform the linearization. The 1D LUT shown in Figure 1-(A) is from an Nikon D40.

Stage 3: Lens/Flat Field correction (Params: 4×ArrayN×M) Many cameras provide a spatially varying correction that compensates for lens distortion and uneven light fall.

Stage 4: Demosaicing (Params: func) The demosaicing step converts the single channel raw image to three full-size R/G/B color channels by interpolating the missing values in the Bayer pattern. We denote this operation as an arbitrary function, func.

Stage 5: Noise reduction (Params: func) Similar to the demosaicing stage, noise reduction is denoted as an arbitrary function, func.

Stage 6: White-balancing and color space conversion (Params: Two 3×3 matrices) This stage performs the necessary color space conversion between the camera specific RGB color space and a standard color space (e.g. CIE XYZ or ProPhoto RGB).

Stage 7: Hue/Sat map application (Params: 3D LUT) This optional procedure is intended to be part of the color space conversion to allow a non-linear transformation to be incorporated to improve the color rendition. While this is referred to as a ‘hue’ and ‘saturation’ modification, it is implemented as a 3D LUT applied directly to the RGB values obtained in Stage 6.

Stage 8: Exposure compensation (Params: EV value, 1D LUT) The exposure compensation is a digital exposure adjustment. While the input is given as an exposure value (EV) that is used to control shutter and aperture settings on a camera, in the digital case, this simply applies a linear gain (either up or down) to the intensities values.

Stage 9: Color manipulation (Params: 3D LUT) Cameras often apply their own proprietary color manipulation that is linked to different picture styles on the camera. Like the Hue/Sat map, this is applied as a 3D LUT where RGB values are interpolated based on the table’s entries.

Stage 10: Tone-curve application (Params: 1D LUT) A camera-specific tone-map can be specified. This is part of the photo-finishing process on board the camera.

Stage 11: Final color space conversion (Params: 3 × 3 Matrix) This color space conversion converts the internal camera working color space into the final output-referred color space. This is done using a 3×3 matrix and is assumed to be related to color space used at stage 6. The most common color space for cameras is the standard RGB (sRGB) and Adobe RGB.

Stage 12: Gamma curve application (Params: 1D LUT) The final stage is a gamma curve that is applied as a 1D LUT with 4096 entries.

These twelve steps make up the collective stages that can be controlled via API calls or direct image modification to intermediate images. Access to this suite of components provides a comprehensive means for manipulating the image from the input raw to its final sRGB output.

3 Conclusion
This paper has presented a new software platform that allows low-level access to the individual components in the camera imaging pipeline. Specifically, our platform leverages the Adobe Digital Negative (DNG) image file format and makes the necessary modifications to the available DNG SDK to provide an extensive API for modifying the parameters of the pipeline, as well as allowing access and modification to intermediate images that can then be inserted back into the pipeline to compute the final output that would be obtained on a camera.

In the poster session the usefulness of this platform will be demonstrated on a number of examples, including white-balance, noise reduction, and colorimetry.

References
Simplification for Large-Scale Fabrication
Michelle Arkhangorodsky† Yanjun Jiang†
University of Toronto

ABSTRACT
3D printing does not scale to large objects due to its restrictions in size, as well as high time and material costs. This project aims at addressing this problem by simplifying an arbitrary triangle mesh into a small collection of 3D-printable joints and connecting rods. Our approach is based on mesh segmentation where regions are initialized by coarsely clustering faces with similar normal vectors. The regions are merged into one another greedily based on distance costs between adjacent regions. After each merge, a preview of the fabricable model is presented using edge straightening. The merging process continues until the user is satisfied with the final result.

1 INTRODUCTION
3D printing makes it possible to convert 3D models into physical equivalents. However, it does not scale to large objects, as the size of a 3D-printed object is restricted to the volume of the printer, the material cost is high, and the printing time is long [1]. The idea for this project is to simplify the input mesh to make it fabricable using joints and rods. Only the joints need to be printed which will significantly save the printing time and cost, and solve the scalability issue. Connecting rods can be made from cardboard, wood, or metal. It is believed that this work can potentially benefit a variety of applications such as theatre props, puzzles, and teaching models.

2 RELATED WORK
Our review of related work mainly focuses on previous approaches to fabrication. [1] aims to approximate a 3D mesh with a small number of planar polygons. After choosing seed faces, their algorithm iteratively switches between segmentation and flattening steps. Their output is restricted to a collection of planar polygons largely to enable texture transfer. Their approach does not require such a restriction. In this way, our task of approximating a 3D mesh with joints and rods is more closely related to [3]. Their approach considers a single node with 62 different slots and selects struts from a set of pre-specified lengths and shapes as specified by the Zometool system. Their approach focuses more on topology preservation under the Zometool constraints as opposed to mesh simplification to a very small collection of joints and struts which is what we will focus on more in this project.

3 SIMPLIFICATION ALGORITHM
3.1 Overview
Our approach can be divided into the following steps: 1) preprocessing an input triangle mesh to a coarse mesh, 2) clustering mesh faces into ‘normal sets’, 3) iteratively merging normal sets (or regions), and finally 4) edge straightening. Edge straightening is done for visualization purposes at the end of every merge iteration. In our program, a salient-face-painting UI is also provided to to enable the selection of regions that are perceptually important to the user, but do not contribute significantly to the topology of the original mesh.

3.2 Normal Set Clustering
To obtain a coarse clustering of the faces of the input mesh, we use a greedy approach to create ‘normal sets’ which are sets of faces with similar normals. A normal set is initialized with a random seed face. Breadth-first search is run from the seed, adding neighboring faces if the angle between their normals and the normal set’s representative normal is below a certain threshold. This threshold is set to be a small angle, as this clustering phase is meant to produce a coarse clustering of the mesh. When a new face is added to a normal set, the set’s representative normal is updated by taking the weighted average of the new face’s normal and current representative normal.

When the search encounters a face that meets the normal threshold, but has been selected as salient by the user in the optional painting step, that face is not added to the currently growing normal set. However, the salient face’s normal vector is used to update the current normal set’s representative normal vector in order to be resistant to unreasonable user input.

3.3 Region Merging
Region merging aims at further grouping clusters generated in the previous section into reasonable regions, whose edges after straightening will become the frame for the physical output. To achieve this, a region adjacency graph (RAG) is constructed, where nodes are the normal sets from the previous section. Adjacent normal sets are assigned a weighted edge, where the weight (distance) measures the cost for cutting such edge to merge the two clusters $i$ and $j$. The Cost$_{ij}$ is shown as follows and three factors considered, $A_{ij}$, $P_{ij}$, and $N_{ij}^2$, are described in details in following subsections. The region merging will then be conducted greedily by first merging adjacent clusters $i$ and $j$ with the smallest Cost$_{ij}$.

\[
\text{Cost}_{ij} = A_{ij} \cdot P_{ij} \cdot N_{ij}^2
\] (1)
3.3.1 Similarity of Representative Normals $N_{ij}$

Face clusters that share similar representative normals can be simulated by a new planar surface without losing too many salient features compared with those clusters that have very different representative normals. Therefore, we keep considering normal similarity in our distance cost as follows:

$$1 - \frac{\vec{N}_i \cdot \vec{N}_j}{||\vec{N}_i|| \cdot ||\vec{N}_j||} \quad (2)$$

where $N_i$ and $N_j$ represent the representative normal of adjacent node $i$ and $j$. The dot product of the two normals is subtracted by 1 to avoid negative values. Dissimilar normals have resulting values of over 1, while more similar normals have values between 0 and 1.

3.3.2 Area of Adjacent Regions $A_{ij}$

The area of adjacent clusters should also be considered in the distance cost to avoid very small regions in the merged result, as they generally contribute little topology but introduce as many joints and rods as larger regions [2]:

$$\begin{cases} \epsilon, & \min(\text{area}_i, \text{area}_j) \cdot \text{area}_{\text{min}} < \text{Area}_{\text{min}} \\ 1, & \text{else} \end{cases} \quad (3)$$

where $\text{area}_i$ and $\text{area}_j$ represent the area of adjacent regions $i$ and $j$. If the smaller area of the two regions is below $\text{Area}_{\text{min}}$ then the area weight $A_{ij}$ will be set to be a very small value $\epsilon$ which encourages merging; otherwise, it will be set to 1. $\text{Area}_{\text{min}}$ is set to the ratio between a small fraction (25%) of the expected number of faces per region and the total number of faces and it is adaptive to the number of target regions.

3.3.3 Shared Perimeters of Adjacent Regions $P_{ij}$

The length of the shared boundary between two adjacent regions is also considered in the distance cost, since regions with longer shared boundaries are more likely to be grouped together compared with a shorter shared boundary. The perimeter weight $P_{ij}$ is computed as follows: [2]:

$$\frac{\min(\text{perimeter}_i, \text{perimeter}_j)}{\text{sharedLength}_{ij}} \quad (4)$$

where $\text{perimeter}_i$ and $\text{perimeter}_j$ represent the perimeter of adjacent regions $i$ and $j$ respectively and $\text{sharedLength}_{ij}$ represents the length of the shared boundary of regions $i$ and $j$.

3.4 Edge Straightening

To convert the mesh from a collection of regions to a set of vertices and edges, we use the edge straightening method described in [4]. This method simply finds the shared boundary between two neighboring regions, and connects a straight line between the endpoints. In terms of fabrication, the endpoints comprise our 3D-printed joints, and the edge between them represents a straight rod. This method is chosen because it will not introduce self-intersections or create disconnected components, which would result in a final simplified object with disconnected components.

4 Experiments & Results

We tested our approach on several meshes and the simplification results are shown in Figure 2. The stopping condition of procedure depends on when the user is satisfied with the simplification, so we experimented with how far we can reduce an input mesh in terms of the resulting number of regions, joints, and connectors, before the result started to become uninterpretable. Furthermore, a 1.5-meter tall real wooden bunny with rods and joints are constructed to demonstrate the fabricability of our method (Figure 3).

5 Future Work

Future work includes modifying our proposed distance function to further to consider how merging two regions will affect the result after edge straightening. Merging two regions greatly increases the likelihood that their original endpoints will be lost after edge straightening. The penalty of losing those vertices can be computed as an approximation of loss in overall curvature or Hausdorff distance to the original mesh, and can be incorporated into the cost.

With respect to fabrication, a post-processing step could be added to make connectors as similar in length as possible. This would reduce the time and effort spent cutting connectors to size, as one cut could be used to create connectors of the same length. Further, the stability of the object should also be taken into account to ensure that the end result is able to stand on its own. Lastly, very small angles between connectors should be avoided as they lead to overlap in the eventual cavities of the 3D printed joints, making them unusable.

6 Conclusions

We have developed a method to simplify an arbitrary triangle mesh into a small collection of joints and connecting rods such that the result retains the interpretability of the original model. We have proposed an interactive approach based on mesh segmentation that does not constrain the resulting object to be composed of planar polygons so that users who are interpreting the result are not limited to perceiving flat, planar objects.

Acknowledgments

The authors wish to thank Alec Jacobson and Sarah Kushner, from the DGP group at The University of Toronto.

References


Figure 3: The final fabricated bunny, Chopsticks.
Rigless Skinning for Interactive Vector Animation

Darren Moore*  Alec Jacobson†  David I.W. Levin‡

University of Toronto

Figure 1: Animation workflow: 1. User draws or imports an image. 2. User skins the skeleton by painting on physical properties. 3. User animates image by applying forces.

ABSTRACT

We present a tool for authoring, rigging, and live-animating 2D vector illustrations that contain secondary motions which add life and expression to the art. Our tool emphasizes seamless transitions between sketching assets and animating them, adding a dimension of interaction to exploratory sketching. These seamless transitions also allow authors to create and perform animation live. This is accomplished with a painting-based skinning interface that provides a quick and intuitive way to define the geometry and physical properties of a sketch. Such properties are represented by different colors that can be painted on, leveraging users’ mental models of painting apps. The painted properties are used in physical simulation to automatically add expressive secondary motions to the animation. Authors are provided several ways to interact with the art such as using spring-based pulling or flicking, and adding forces like gravity or wind.

Index Terms: Human-centered computing—Human computer interaction (HCI)—Interaction paradigms—Graphical user interfaces;

1 INTRODUCTION

To create lifelike and expressive 2D animation, artists add secondary motions and deformations that go beyond the primary translations or rotations. Such secondary motions are arduous to add by hand, so physical simulation can be employed to add these motions automatically. Unfortunately, this requires a rig and explicit representation of the object’s geometry, which is also difficult to create.

We introduce a tool for seamlessly authoring, skinning, and animating vector art. An intuitive sketch and painting-based interface allows novices to quickly and easily create expressive performed animation, adding another dimension to sketching and doodling. The main contributions of this work-in-progress are: 1) An intuitive painting-based interface for easily skinning 2D illustrations with geometry, and 2) A framework for bringing illustration and interaction together seamlessly.

* e-mail: da.moore@mail.utoronto.ca
† e-mail: jacobson@cs.toronto.edu
‡ e-mail: diwlevin@cs.toronto.edu

2 RELATED WORK

The techniques of Energy Brushes [6] and Draco [2], easily bring drawings to life with motion, but are more suited to cyclic or particle-based motions. Motion Amplifiers [3] includes methods to stretch and deform a larger class of objects, but none of these three are designed for the final result to be interactive.

Kitty [1] allows users to create functional relationships between objects, leading to meaningful interactions, but does not include methods to add life-like secondary motions to the objects. Further, Kitty is not designed to be used to create live animations, as interactions must be predefined with its graph-based interface.

Willett et al. [5] explores adding secondary motion to live, performed animation. It allows users to create different types of rigs for their art, then animate them with a plugin to Adobe Character Animator. While this allows for interaction and live performance, there is a large divide between authoring the art, and rigging and animating it. We seek to close this gap, allowing seamless artistic exploration.

3 SYSTEM DESIGN

The interface (Fig 2.) consists of a Canvas for sketching, Tools panel, Layers panel, and context sensitive Properties panel. These were designed to be easy to understand by leveraging users’ mental model of painting apps like Adobe Photoshop and Adobe Illustrator.

Figure 2: Our user interface, consisting of (a) a canvas, (b) tools panel, (c) layers panel, and (d) properties panel.
Users begin by using the pencil tool to create their artwork. Since the system uses a vector art representation, authors can use this to their advantage by drawing with not just stroked lines, but fills, dashed lines, and other brushes. After creating the artwork, users must define the geometry and physical properties.

We note that the artwork’s geometry can’t be automatically extrapolated from the art alone, since there are many ambiguous cases where the geometry depends only on artist intent (see Figure 3). The user may use the physics brush to paint on their intended geometry and physical properties. The process of doing this mirrors how an artist might color-in an area with a brush in any digital painting application, making skinning the art fast and simple.

Users may select from preset materials or can choose custom physical properties. Exposed properties are Younge’s Modulus (we call "rigidity") and mass. Areas can be painted to be anchored as well.

Once the artwork has been skinned, users may interact with it in several ways. We provide:

1. A pulling tool that attaches a spring to the geometry, allowing users to drag objects around. See Figure 1, where a user pulls the skeleton’s arm to make him wave hello.
2. A pushing tool for users to flick objects and see a response. For example, users may feel inclined to flick the long nose of the character in Figure 4.
3. A tool to translate objects around while they exhibit secondary motion.
4. Gravity defined by an accelerometer. By shaking the drawing tablet, users can see a response in the art they just drew.
5. Wind, by applying cyclic forces. Figure 4 includes a stoic oak tree, waving in the wind.

Even while the art is in a deformed state, users may re-select the pencil tool and continue adding details or use the physics brush to redefine the physical properties. The ability to seamlessly switch between authoring and interacting gives users a chance to have more fun actively sketching instead of wrestling with multiple programs.

4 IMPLEMENTATION

Strokes are stored as cubic Bézier spline curves which are skinned with the method of Liu et al. [4]. As the background grid deforms, the art moves correspondingly. This method transforms the art while preserving its vector nature, hence exported art can be edited as vectors in any vector editing application or reimported by another user to animate.

Physical simulation is done using meshfree particle methods and a linear elastic material model, though our technique is independent of material model and may work with finite element methods.

5 CONCLUSION

To avoid the difficulty of rigging or creating animation keyframes, we introduce a painting-based skinning interface that leverages users’ mental model of painting to assign physical properties to any piece of art. This approach gives users control over the stiffness and mass at any part of the artwork. Our seamless workflow adds another dimension to sketching and doodling, that of interaction and motion.

REFERENCES

The oil and gas industry has long used visualization to help analyse tasks performed. One key task is to inspect embedded structures such as layers, relative spatial location to wells or faults, and the form and location of specific geological features that effect production. High-impact decisions are made as a result of the analysis of these datasets, often with time constraints and in collaboration with a variety of stakeholders across disciplines. Often in real-world scenarios, collaboration must occur between users with different levels of access to the data. Some users such as decision makers and key technical personnel within a government agency or oil company will have privileged access to the data. Others such as sub-contractors, trainees or partners in joint exploration ventures may have a lower level of access to the data. Currently, in order to collaborate over this data, all parties will be shown this sensitive data even if it is not required, relying only on legal agreements to prevent information leakage. Alternately manual effort will be expended to create carefully sanitized datasets that have important details removed. The sanitized models are then provided to parties with lower privilege and then conventional collaboration methods such as meetings are used to discuss these models with higher privilege users that have greater insight into the data.

There are significant flaws with both of these existing approaches. When sharing highly sensitive information with only legal protections, the fundamental security guideline of providing the minimum amount of information to parties that require it is not followed. Certainly, legal agreements can mitigate loss of information, however there is still no complete assurance that there will not be information leakage. All users have a complete understanding of the data, and it may be difficult to prove how that information is disseminated should information leakage occur. When sanitizing data and collaborating over the data using conventional techniques, much time and effort may be spent properly sanitizing the data, and errors may be made. Collaborating between users with different levels of access to the data that do not have a real time view of the specific information can be time consuming and frustrating.

Our system proposes to solve both of these issues. Users that have a lower level of access are provided a view that shows only what they need to see, and all parties can enjoy the efficiencies of real-time collaboration over the same dataset.

2 Related Work

Augmented reality (AR) security has been discussed broadly in a variety of works such as [6]. In this work security and privacy challenges for AR have been organized along two axes: system scope and functionality. Functionality is categorized into output, input and data access. The authors consider these functionality challenges as they arise in three different system scopes: single applications, multiple applications and multiple systems. Examples of these challenges chiefly relate to the sharing of input and output devices in real time and the complex access controls that are required on sensor data. The authors note however that this is a relatively unexplored medium that despite the challenges, presents many opportunities to address security and privacy in novel ways.

A variety of visual protection mechanisms have been applied to maintaining the privacy of individuals that are recorded in video [4]. There have been some attempts to introduce visual obfuscation to help users focus on important information while de-emphasising information that is less important in a given context [3]. However
there seems to be little work that applies visual protection methods to facilitate collaboration over data by users with different levels of access to the data.

3 Collaboration Tool

The current implementation supports collaboration between the HTC Vive, Oculus Rift, Microsoft Hololens and a public display. There is a simple representation of an avatar for each collaborator, and users can work together on separate devices. Collaboration may also be achieved in our CAVE. We will often take work from HMD devices and display it in the CAVE for larger presentations or collaboration sessions.

Our prototype does not only allow binary access to the data. Users may be denied access and provided full access. Users may also have a third level of access which we call conceptual access. This provides a conceptual overview of the information while reducing the detail available. This is very helpful for collaboration since a user with a higher level of access can collaborate with users with lower levels of access, when a conceptual view of the information will suffice. In our application specifically, users with conceptual access are able to view the structure and spatial location of features, but are not able to see the detail such as the property values of the cells or the colour of the cells that represents their property values. This allows a user with a higher level of access to see all the information, while providing only structural and spatial information to users with only a conceptual view. Areas of key information may also have a protected region placed around them that prevents viewing of any detail so neither properties or structure is visible.

Figure 1: Detail and context views. Reservoir scenario view around the well with cells removed for inspection of interior details. On the left is the view from the HTC Vive with full access to the protected area. On the right is the view on the Microsoft Hololens with the property values obscured so the user can only see the structure.

Figure 2: Detail view with avatar and protected area. On the left, view of the reservoir on the HTC Vive with full access. On the right is a the view of a protected area using the Oculus Rift where the user has no access.

In an initial evaluation with three reservoir engineering subject matter experts, we received very positive feedback, validating that providing an asymmetry of information in a real-time collaboration tool was useful and would help them work with varied parties more effectively. When performing a simplified but representative reservoir engineering well placement task they were able to achieve this task by working together, despite some participants only having a conceptual view of some areas no access to other regions. Although these results are preliminary, this provides important guidance and validation by subject matter experts with an academic and professional background in this field.

4 Future Work

We hope to run further studies to identify user preferences and needs related to protecting information in these collaboration scenarios. As we gather more insight into user needs and preferences, we hope this will provide us with additional methods of visual obfuscation that will protect against information leakage while enabling collaboration. We hope that we can find additional ways to apply these protection mechanisms that go beyond demarcating a spatial region. Finally we would like to perform more studies to validate the techniques that we have currently implemented.

5 Conclusion

A workplace environment can have complex privacy requirements and diverse user groups with different backgrounds and expertise. Data must be protected in scenarios that may require collaboration with social or time pressures encouraging bad privacy behaviour on the user’s part. We have analysed the needs of industry experts when collaborating over sensitive reservoir engineering data. Based on this analysis, we have developed a tool that directly addresses this need. Based on a feedback session with subject matter experts, we have received some preliminary but overwhelmingly positive feedback regarding our visual obfuscation approach to controlling information leakage in a collaborative immersive reservoir engineering scenario.

Acknowledgments

This work was supported by the Energi Simulation / Frank and Sarah Meyer Collaboration Centre.

References

Using Supernumerary Robotic Arms for Background Tasks

Anna Tran, Sowmya Somanath, Ehud Sharlin
University of Calgary, OCAD University

ABSTRACT
Numerous studies have envisioned the explicit and implicit use of Supernumerary Robotic Limbs (SRLs), wearable robotic limbs, to directly assist a user in performing tasks. In this paper, we explore in which situations Supernumerary Robotic Arms (SRAs) could be used in to perform background activities. We conducted a preliminary design study to better understand user expectations for using SRAs in background tasks. Our results highlight that SRAs can be helpful in performing background tasks alongside users performing a primary task. Informed by our study we present our current implementation efforts and suggest directions for future work.

Keywords: Supernumerary robotic limbs, human-robotic interaction, background activity

Index Terms: H.5.2. [Information Interfaces and Presentation]: User Interfaces

1 INTRODUCTION
The notion of cyborgs, “part human, part machine” entities, is beginning to move from the literary world to the physical one. Disabled individuals with prosthetic limbs is a real-world example of today’s cyborgs [1]. Research has revealed that prosthetic limbs have significant potential to positively impact quality of life and daily usage. For example, a commercial wheelchair arm JACO developed by Kinova [1,2], although not biologically infused with the user, can help users with activities of daily living.

In response to these findings, many research projects have proposed and prototyped Supernumerary Robotic Limbs (SRLs), a human-wearable, yet detachable system composed of mechanical limbs capable of augmenting the user’s physical capabilities [3-8]. Using SRLs, humans can learn to adapt to a cyborg mentality without the commitment of changing their physiological structure to do so. SRLs offer some of the same advantages as their biological counterparts: examples include bearing physical loads borne by the user and helping the user with grasping objects [4,6,8].

Previous studies have highlighted various instances where SRLs could be used for primary tasks, namely in medicine and manual labor. However, very few of these studies have actually generalized to the use of SRLs in background activities. Our overarching long-term goal is to explore the types of tasks SRLs can help with, and how a user controls these limbs to carry out a task. Based on the examples presented in literature [3-6], we posit that SRLs can help with two types of activities – primary and background tasks – and within these activities, SRLs can be controlled either explicitly or implicitly. Primary tasks are often the focus of the user. Background tasks are not the focus of the individual but are still supplementary to the main activity. For example, in construction work, building an artifact is the primary task and handing someone tools is a background task.

In this work-in-progress project, we propose to explore the use of implicitly controlled SRAs for performing background tasks. We imagine that humans might prefer to carry out primary tasks themselves. As a result, SRAs may be beneficial for performing background tasks supplementary to the user’s primary task. In the remainder of the paper, we present the current state-of-the-art research in this realm, and then present our exploration and findings of possible background tasks that SRAs could perform to augment primary tasks.

2 RELATED WORK
Several researchers have explored the use of SRLs for specific applications. For example, physical rehabilitation has motivated the construction of self-learning and user-controlled Supernumerary Robotic Fingers, devices meant to assist humans in primary activities such as grasping objects [6,8]. These extensions of the hand have enhanced basic grasping behaviours, what some researchers consider the most fundamental functional requirement for manipulation [6].

It is clear that SR Fingers can be quite useful in accomplishing primary tasks but having SRAs execute background tasks can be equally beneficial. One study conducted with human participants explored how SRAs can be useful in performing a coordination-based task; in this case, stabilizing a workpiece as a construction worker drilled into it [3]. Another design concentrated on a bracing strategy used by SRAs to reduce static loads borne by users completing physically demanding tasks [4]. While both of these cases underline the effectiveness of SRAs in supporting users as they focus on a primary task, the SRAs were only simulated by two human actors and have yet to be implemented.

A few studies have managed to produce SRL prototypes capable of assisting humans with both primary and background activities. Sasaki et al. experimented with the adaptation of humans to “MetaLimbs,” SRAs whose function was explicitly mapped to the human foot [5]. In a series of tests, participants learned to use robotic arms for both primary and background tasks ranging from grasping cups to picking up phone calls while typing [5]. A mechanical hand augmentation device was developed by Leigh and Maes for the purpose of studying “synergistic interaction” [8]. Their device assisted users in performing two-hand operations using a single augmented human hand.

This project builds on and adds to the current body of work by exploring the use of SRAs from a unique perspective. Unlike earlier studies, the goal is not to create a new design for SRAs but to examine the interactions between humans and implicitly functional SRLs used in background activities. It focuses on a larger scale than just hand augmentation devices, specifically robotic arms, which can move more independently of the user’s movements.

3 PRELIMINARY DESIGN STUDY
We conducted a preliminary design study to research what users would expect from additional robotic arms for performing background activities. For this study, to familiarize the
participants with the idea of SRLs and background activities, we selected a few simple activities such as a colouring task and a selfie-taking task as starting points. We chose these tasks as they clearly demonstrated implicit control of the SRAs.

Similar to the methodology used by [3], we conducted our preliminary study using human actors. Our preliminary study involved three pairs of human participants. From each pair, one would simulate the SRA and the other would be the end-user. Having both roles played by humans provides insight on two important perspectives: how participants imagine SRAs would work around the user, and how the user would perform an activity in collaboration with SRAs.

Participants first took part in the colouring task (Figure 1a). The taller of the two participants would act as an SRA and the shorter of the two would be the human user. The user was instructed to draw five objects they would find in a home. Once the user had completed a drawing, the participant simulating the SRA would colour in the drawing using pencil crayons.

The second was a selfie task, where each participant had the opportunity to be both the SRA and the human user (Figure 1b). In this situation, the user held up the artwork from the previous task while the arms, handling a selfie-stick with a smartphone attached to the end, took pictures of the user from different angles.

Each pair shared feedback of their experience in a post-study interview. We also asked participants to reflect on other potential background tasks that SRLs can help with. The audio and video recordings of participant responses were transcribed and the similarities and differences between the responses were noted.

We observed a few recurring patterns among the responses from the participants. They agreed that SRAs could be helpful in parallelizing tasks, although there were varying opinions on the types of tasks SRAs should help with. Some imagined that SRAs would only be useful when supporting users with complex tasks normally requiring more than two hands, for example in construction work. Others wanted to use them for simple, mundane tasks like cleaning. They also thought that implicitly functional SRAs should be able to understand and act according to the user’s intentions, while not interfering with the user’s own work. While they were comfortable with implicit control, several participants also wanted some degree of explicit control over the robotic arms. In terms of the physical design, participants expected SRAs to be easily detachable and easy to store when not in use. Of the two design study tasks, participants felt that SRAs interfered less and were more helpful in the selfie task. Thus, we decided on replicating the selfie experiment for the main study.

4 CURRENT PROTOTYPE SYSTEM

Informed by our study, going forward, our proposed implementation incorporates a similar selfie-taking task. A participant is introduced to Baxter, a humanoid robot whose right arm will act as SRAs throughout the experiment as depicted in Figure 2. Attached to Baxter’s right arm is a selfie-stick with a smartphone hooked onto the end. Attached to the robot’s core are backpack straps so that the user can simulate the experience of “wearing SRAs”.

The participant will pose holding an object of choice with both hands. An audio file will be played before taking a picture, to notify the user of the robot’s intended action. The right arm will move based on a timer (rather than explicit user commands) to vary the angle of the camera and take selfie pictures.

5 CONCLUSION

In this paper, we explored human-robotic interaction between humans and SRLs performing background activities. From our preliminary design study, we found that individuals preferred using implicitly functional SRAs for background tasks. Going forward, we plan to improve our implementation wherein the implicit behavior of the SRA is based on specific cues such as body movement of the user, and facial expressions. We also want to incorporate a more elaborate “vlogging” experiment wherein the robot takes short videos of the participant crafting art. Using the more thorough implementations, we plan to conduct further user studies to determine the usability, effectiveness, and desirability of the SRAs for background tasks.

REFERENCES

Mental Health in Online Communities: How University Students Use Online Communities to Discuss Mental Illness

Denise Y. Geiskkovitch, Andrea Bunt, James E. Young
Department of Computer Science
University of Manitoba

Abstract

Many students experience mental health issues while in university. Disclosing such issues, to a counselor, or friends and family can help cope with mental illness. Recent research found that it is also common for individuals to disclose mental health problems on social media. We collected mental health posts made by university students on online communities, and conducted a thematic analysis to investigate what students are posting about, and why they might be posting. We found that forum communities tended to have longer posts, usually disclosing mental health issues and asking for advice, while social media content focused on the negative impact of university on mental health, and lack of mental health support from universities. Replies to forum posts contained advice, while those to social media content offered support or empathy.

Keywords: Online communities, mental health.

Index Terms: H [Information Systems].

1 INTRODUCTION

Mental health disorders are widespread across university campuses. Trying to balance school and work expectations, while managing other life activities can often lead to emotional and mental issues such as depression and anxiety [4]. Previous research has found that disclosing that one has a mental illness can be a way of coping with it [5]. Recently, research has focused on how social media is used to disclose these types of sensitive information, and the responses received from others (e.g., [1, 3, 6]). However, it is uncertain whether social media, and online communities in general, are helpful in coping with mental illness, or mainly used as an emotional outlet. Here, we investigate whether university students use online communities to self-disclose mental health problems, why they may do so, and how these disclosures are received by the communities.

We conducted an exploratory thematic analysis on data collected from online communities to investigate whether university students disclose mental health problems in online communities, and whether the content of their posts sheds any light on their reasons for posting and how it may help them with coping. We specifically targeted online communities in which people can make open posts and others can comment or provide feedback (e.g., Twitter, reddit).

2 RELATED WORK

University students are at a high risk of suffering from mental illness. Almost half (41%) of all university students, at one point or another, may experience symptoms that meet the criteria for a diagnosable mental disorder [7]. In addition, it has also been reported that the number of students with mental health issues appears to be on the rise [8]; more students now report and are diagnosed with illnesses such as depression, anxiety, and bipolar disorder [2].

Recent research has also been investigating how mental illness is disclosed on social media platforms such as Instagram. For example, Manikonda & De Choudhury [6] and separately Andalibi et al., [1] examined mental health on Instagram. Manikonda & De Choudhury [6] examined how images posted on Instagram can disclose mental health issues, and found that there are several clues that we can look for to detect mental illness, such as color and visual cues. Andalibi et al. [1] on the other hand investigated how different types of mental health posts on Instagram receive varying positive and negative responses. They found that some comments were actually negative, and encouraged or supported harmful behavior. This highlights potential risks associated with disclosing mental health issues online.

Another online community in which people share their mental problems is reddit. De Choudhury and De [3] investigated how individuals self-disclose their mental state on reddit. Through linguistic quantitative analyses they examined the kinds of mental health discussions posted, what factors contribute to social support, and developed a language model to characterize support.

In this paper, we contribute to the existing literature by investigating (1) why individuals (in our case university students) may choose to disclose mental illness online (especially when responses can be negative), (2) if they do so on online communities other than social media, and (3) whether these disclosures and the responses received in return are beneficial to those impacted by mental illness.

3 METHOD

We originally planned to conduct an online survey with university students to investigate whether they disclose mental health information online, and how and why they choose to do so. However, initial exploration into this area indicated the need for preliminary research on the ways and extent to which students self-disclose mental health in online communities. This research is a first step in that direction. We conducted a qualitative thematic analysis of mental health posts by university students found in a variety of online communities to observe how interactions take place, the types of responses received, and whether those responses, or the mere act of posting, may help individuals’ mental health.

3.1 Procedure

During three weeks in March 2017, we collected all of the posts from online communities that met the following criteria:

*Poster is a university student* – the post either indicates this explicitly, or implicitly through talk about university or college.

*Post’s focus is mental health* – we only included posts which had mental health issues as their main focus (as opposed to posts that merely mentioned mental health).

*Post is public* – so that we could get access to it.

To find posts we searched a variety of online communities, including Twitter, reddit, Facebook, Tumblr, and online support
and discussion forums found through Google. We found the posts through keyword searches containing a subset of the words “college,” “university,” “student,” “depression,” “anxiety,” “bipolar,” etc. For communities containing tagging systems (e.g., ‘#’), those keywords were also searched for as tags.

We then conducted a qualitative thematic analysis, where we grouped posts according to categories that emerged from the data. We also analyzed the content of the comments, likes, shares, and votes provided in response to the original posts.

4 RESULTS AND DISCUSSION

In a three-week period, we found 74 posts containing personal mental health information or disclosures in a variety of online communities. The majority of posts found were from the Twitter and reddit online communities. No posts were from Facebook and Tumblr, likely due to privacy settings. A small amount of posts was found in other online forum and discussion communities, these posts however had not been posted within the three-week period. We chose to include posts from these communities, as long as they were from the same calendar year, because they did not seem to be different from others found within our time period. Had these posts not been included, the comparison would have only been between Twitter and reddit, and our intent was to compare a number of different communities as long as they possessed the correct content.

4.1 Differences in Online Community Platforms

Several themes emerged from our initial exploratory thematic analysis. One main theme was that there was a fairly distinct relationship between the online communities and the nature of the posts. For example, forum and discussion communities tended to have posts asking for advice and disclosing mental health information, rather than any other categories. Social media content (in this case Twitter) on the other hand, had more posts related to the lack of mental health support from their institution, how university affects mental health, and self-disclosing personal mental health issues.

While social media posts were more humorous, or tried to downplay mental health issues (see Figure 1), and did not usually ask for advice, forum posts were much longer (likely due to an unlimited amount of allowed characters) and more detailed. Most of the forum posts collected contained detailed information about the individual’s mental health condition, and often times relayed much of their life story. In this sense, forum posts also contained much more sensitive, and sometimes disturbing, information (such as self-harm or suicide plans).

4.2 Responses to Mental Health posts

While almost all of the forum posts received comments, only a handful of the social media posts did. Both, however, received other types of recognition, such as likes, sharing, and up-votes (all posts had some type of recognition). All but one forum posts with comments contained at least one comment providing advice, even if the post did not ask for any type of advice. Social media posts however rarely received advice, but more often comments relating to the poster’s situation, some self-disclosing mental health problems, and others offering support. Forum comments sometimes asked for clarification, while social media comments did not.

An interesting observation we obtained from this data is that, except for one forum poster, only social media posters openly acknowledged comments provided by others. Out of all of the posts we obtained, only one forum poster thanked a commenter for their advice and support, while a handful of social media posters thanked their commenters.

While most comments were positive and supportive, some of the comments provided in response to posts were of a negative nature, either questioning the original poster about statements they had made, or sharing a sense of hopelessness about the current situation (see Figure 2). These comments however were only found in reddit posts asking for advice. No other occurrences of negative comments were present in any of the other online communities.

Figure 1. Twitter post downplaying mental health issues.

Figure 2. Reddit comment empathizing with poster.

4.3 Why post on online communities?

There may be a number of reasons why individuals, in this case university students, may talk about mental health in online communities. Our research suggests that asking for advice might be one of the reasons students self-disclose mental health issues on online communities, especially on forums. In addition, online communities allow individuals to hide their identities, so it might be a good place to vent, or get others to interfere (for example, hoping to be convinced to not take drastic measures).

5 CONCLUSION

Many university students experience mental illness, and choose to disclose it on online communities. In this paper we presented our initial investigation into mental health disclosures of university students on online communities. We found that forum content usually disclosed mental health issues and asked for advice, while social media content was not as detailed. This research showcases the need for a formal inquiry into why university students choose to make such posts, and how communities could help support them.

6 REFERENCES


Conveyor: A Dual-Task Paradigm for Studying VR Dialogue Interfaces

Patrick Dubois\(^1\), Daniel J. Rea\(^1\), Kevin Hoang\(^1\), Meghan Chua\(^1\), Danielle King\(^2\), Corey King\(^2\), James E. Young\(^1\), Andrea Bunt\(^1\)

\(^1\)Department of Computer Science, University of Manitoba; \(^2\)ZenFri

**Abstract**

VR applications can enhance players’ sense of presence within virtual environments. A common scenario is to have a player working on a task, while simultaneously making dialogue selections using VR. We investigate a dual-task experiment design for this scenario, and an initial study using four interfaces for dialogue selection. We found that interface naturalness – one measure of immersion – seems to have a large role in player preference, regardless of selection speed.

**Keywords:** Virtual reality, dialogue selection.

1 **Introduction**

One compelling aspect of Virtual Reality (VR) is the immersion factor, where users can experience an increased sense of being present in the virtual world. Immersion has a number of potential user benefits including better emotional responses [7], skill retention [6], engagement [4], or enjoyment [6].

In some VR applications, like games [4], collaborative environments [5], or education tools [6], users can be performing a task in the virtual world while simultaneously conversing with a virtual agent. A challenge, however, is engaging the user in these dialogues without negatively impacting their immersion in the primary task. For example, traditional dialogue-selection interfaces may require diverting a user’s attention from their main task in the virtual world. They might also involve input methods that are being used to complete main tasks, increasing cognitive demands and breaking immersion.

In this work, we investigate different selection mechanisms for dialogue-based interaction and their impact on tasks completed in a virtual environment. Techniques can leverage potentially intuitive 3D interactions, such as pointing, or gaze based interfaces, but can be challenging due to the imprecise nature of human movements [2,3]. Traditional menu and selection designs often result in equal or better performance to 3D designs, but they may break task immersion [1,2]. As our interest lies in both immersion, and performance for dual-task VR situations, we explore both natural, 3D interactions, as well as more traditional selection techniques. As a first step, we investigate a dual-task experiment design for VR with flexible difficulty, and conduct an initial exploration of conversation-picking interfaces in our dual-task VR environment.

2 **Dual-Task Design for Conversation Interfaces in VR**

Our dual-task experiment design had two main goals: 1) to provide a main task that is difficult and engaging enough such that the secondary, conversational task could impact both task performance and engagement, and 2) to have a conversation where user engagement is important, so users take care picking their dialogue.

---

{patrick.dubois, daniel.rea, young, bunt}@cs.umanitoba.ca, 
{hoang, umchua2}@myumanitoba.ca, 
{danielle, corey}@zenfri.com
Trigger: Virtual switches are placed in front of the user, and are activated by pressing a button. 

Head Gaze: Gaze is used to point to a conversation option, with a controller button to confirm selection. 

Voice Recognition: Options are read aloud. This was implemented with Wizard of Oz in this initial exploration. 

3.2 Dual-task implementation

Participants completed four choose-your-own adventure stories, each with a different selection technique. Each story was approximate five minutes long, and was created by a professional video game story writer. Each story was narrated by a different person: two female, two male. Dialogue options were displayed to the user as text on a virtual board when the stories branched (Figure 1). The user would use an assigned technique to select an option.

Users were seated in front of the conveyor belt and could choose both hands, which held the hand-tracking HTC Vive controllers, and their head, wearing the HTC Vive headset. Players had to pick up objects as they passed, by intersecting the virtual controller with an object and holding a button, moving the object to the correct bin, and dropping the object. Objects disappeared if they reached the end of the conveyor belt or fell to the floor.

3.3 Measurements

Our primary performance measures were: time to select conversation options, time to complete each story, number of sorting errors, and number of correct answers to post-task comprehension questions. Immersion was measured via a standard VR immersion questionnaire [4], which we modified to fit our specific VR application.

3.4 Procedure

Participants were introduced to the sorting task, the equipment, and the dialogue selections. This was followed by a practice sorting task, where participants could sort shapes for two minutes, without needing to select dialogue.

Each selection interface was explained before it was used, followed by another two-minute practice session with both tasks. This was followed by the real task, then the story’s comprehension question, and post-condition questionnaire.

At the end, participants filled in both a demographics and overall experience questionnaire, where they ranked the interfaces in their order of preference and gave general feedback.

4 RESULTS

A repeated-measures ANOVA (Figure 2) found a significant effects interface type on self-reported story engagement (F=3.6, p<.05, \( \eta^2=.25 \)). Post-hoc tests, however, were not statistically significant. We also found a main effect of interface on how natural it felt to select a dialog option (F=4.5, p<.01, \( \eta^2=.29 \)). Post-hoc tests found a trend for Voice to feel more natural than Touchpad (p=.07, mean difference of 1.4 ranks, 95% CI [.09, 2.9] ranks) and Gaze (p<.05, mean difference of 2.1 ranks, 95% CI [1.3, 4.0] ranks).

For performance metrics, we found a main effect of interface on dialogue selection time (F=30.7, p<.001, \( \eta^2=.74 \)). Post-hoc tests found Voice slower than all interfaces, (p<.001, mean differences of 1.4s, 1s, and 2s, 95% CI of [.09s, 2.9s], [-.42s, 2.4s], and [.13s, 4.0s] for Touchpad, Trigger, and Gaze respectively). All other tests were non-significant, and we observed large variances.

For preference data, Voice had seven favourite rankings, compared to three for trigger, and two for touchpad. No participant ranked Gaze as best, and no participant ranked Voice as worst.

Figure 2. Significant results. Time to select is in seconds; others are rank. **p<.01, *p<.05, ***p<.001. Errors bars are 95% CI.

5 DISCUSSION AND RECOMMENDATIONS

Our results found higher ratings for involvement and naturalness for Voice, lining up with our interface-rankings data. This may not come as a surprise: users did not need to use any specialized controls to make their selections, and could keep sorting using both hands. However, it was the slowest to make selections, as Voice takes time to correctly read the option. This may imply that speed of entry was not considered important by participants in our task.

Gaze was also surprising: while users had the fastest selections, they rated it poorly in terms of preference, story involvement, and naturalness. It is possible that this is due to a “fat-finger problem” as our targets may have been small when compared to the average head jitter. Gaze may also be inappropriate in dual-task situations, as users’ visual attention was needed for the main task.

We saw a lot of variability in performance data, implying that our sorting task might need refinement. We selected a sorting task because there are many parameters we could manipulate to control difficulty. However, despite extensive piloting, our final configuration seemed too easy for our users. This implies it is non-trivial to find a difficulty that requires some degree of the participant’s attention, but not to the point they could not pay attention to the story, or feel discouraged. Methods to calibrate this difficulty is an open challenge.

6 CONCLUSION

We presented an initial testbed for a dual-task VR application, where a player needs to complete a primary task while making dialogue selections. We used this environment to investigate immersion using four different dialogue-selection methods, and found that study participants seem to prefer methods that allow greater immersion than efficiency. We plan to refine our scenarios to improve our understanding how selection interface impacts player immersion and task efficiency.

REFERENCES


Adaptive Visual-Diagnostic Training: User Mental Model Development

Sarah Hoven1, Alexis Seniuk1, Michel Martel1, Hussain Khan1, Carrie Demmans Epp1, Liam Rourke2
Department of Computing Science1 and Faculty of Medicine and Dentistry2, University of Alberta

Abstract
Current diagnostic imaging identification techniques for skin lesions fail to address the visual nature of this task and do not provide information about the mental models users develop after receiving training. To address this issue, the MSKBLOCK medical training system uses a visual approach to support the development of a user’s mental model. Our study evaluates the usability of this adaptive system and how it affects the establishment of users’ mental models. Those using MSKBLOCK developed mental models based on visual information and learned to distinguish between malignant and benign skin lesions, suggesting the potential for reformulating medical training using technologies that align with the visual nature of some diagnostic tasks.

Keywords: Medical training, e-learning, visual cognition, mental models

Index Terms: H.5.2. Information interfaces and presentation (e.g., HCI): User Interfaces

1 INTRODUCTION
Skin cancer is one of the many life-threatening diseases we can get. Like with other forms of cancer, early detection is important for positive outcomes. Diagnostic imaging is widely used within medicine [1], and it is one of the ways in which skin cancer can be identified. However, interpreting the images accurately and consistently is difficult because the difference between a normal (benign) and abnormal (malignant) skin lesion, in these complicated visual stimuli, is subtle. To train doctors in learning to differentiate between normal and abnormal lesions, medical educators employ a variety of approaches that have been shown to be ineffective [2], at least partly, because they frame this task as a cognitive one that draws on declarative knowledge and clinical reasoning rather than a visual task that draws on perceptual expertise [3].

To address this gap in medical training, an adaptive system (MSKBLOCK) was created to engage trainees in visual categorization tasks. Thus tuning their visual systems to the features of diagnostic images that distinguish normal from abnormal results [4] instead of training the declarative components of image interpretation, such as facts about the biological bases of disease. In contrast to MSKBLOCK, recent technologies have been developed to enable expert diagnosticians to communicate their existing mental models by annotating images [5] rather than developing their diagnostic skills.

When technologies aim to support medical student training, they tend to focus on supporting collaborative diagnostic processes [6], the acquisition of physical skills [7], or the use of technology in surgical settings [8]. In contrast, MSKBLOCK uses a new approach to developing medical trainees’ diagnostic skills when the task is inherently visual: this approach employs adaptive visual training that differs from the commonly employed problem solving approaches of other adaptive learning systems. Evaluations of MSKBLOCK’s effectiveness have been consistent with previous research on visual expertise: that is, this type of discrimination training can lead to quick changes in performance, cognition, and neurophysiology [9]. While these improvements in performance have been studied, the mental models that users develop after using MSKBLOCK and system usability have yet to be studied.

Keeping the above limitations in mind, we conducted a study both 1) to evaluate the usability of MSKBLOCK and 2) to determine the mental models that users developed of skin lesion classification following MSKBLOCK use.

2 METHODS
To achieve the above goals we conducted a usability study and analyzed the development of users’ mental models of skin lesion classification. This design-based research combined interviews with a think-aloud protocol to collect data so we could better understand how users responded to this new training approach and later improve the system.

2.1 Participants
A convenience sample of 8 people (5 female, 3 male) were recruited. Of these people, 3 were between 18 and 24 years old. The other 5 were between 25 and 34 years old. Following oral consent, they used MSKBLOCK and completed their participation.

Six participants had some type of post-secondary education, one chose not to disclose his level of education, and the other had earned a high school diploma. The participant who withheld his educational information, was the only one who did not say he was comfortable using computers or web applications. All others expressed that they were comfortable using these technologies: they selected agree or strongly agree in response to the following statements: “I am comfortable using a computer” and “I am comfortable using web applications”.

2.2 Study Instruments and Procedures
This study consisted of 2 phases of system use: a training session and a testing session. Each phase was book-ended by an interview.

These interviews had structured and semi-structured components. With the exception of participant demographics, the structured components used a 5-point Likert scale (1 - strongly disagree to 5 - strongly agree) to collect participant responses to statements about their comfort with different types of tasks, their experiences using MSKBLOCK, and their confidence in their ability to distinguish between harmless and suspicious skin lesions. The semi-structured components of all three interviews asked participants to describe the differences between suspicious and harmless skin lesions. The final interview collected additional information about how the materials within MSKBLOCK may have challenged participants’ existing ideas about different types of skin lesions. This final interview was also where we collected information about the usability of key system features: Likert-scale items and open-ended questions were again used to collect data about participant experiences.

The two phases of the study were conducted using a concurrent think-aloud approach, which was supported by the speech communication protocol [10]. Under this protocol, we used acknowledgement tokens like “mm hmm” and probes like “and now...?” to encourage the participant to continue to voice their

* The first and second author contributed equally.
{shoven, aseniuk, martell, hkhan, cdemmansepp, lrourke}@ualberta.ca
thoughts with minimal disruption to their cognitive processes. For participants who were not articulating their thoughts after we attempted to stimulate a response, we intervened by asking questions like: What are you thinking? or How close was that to what you expected? [10]

2.3 Data Analysis
Observer notes from the think-aloud sessions and participant interview responses were combined and analyzed to determine the mental models that participants had developed. We were able to track the development of these models over time because the three interviews collected information about the visual features that were important for distinguishing between abnormal and normal skin lesions. In addition to tracking mental model development, participants’ mental models were triangulated across participants and observers. We also performed member checking to ensure that we accurately recorded participant perceptions. Descriptive statistics are used to report responses to Likert-scale items.

3 RESULTS
3.1 Usability Evaluation
In terms of usability, the participants found, in general, that the instructions were sufficient to complete the required image categorization tasks in MSKBLOCK. The users were confident navigating through the question sessions. However, they suggested that the system would be better if the session progress was more prominent on the page.

3.2 Skin-Lesion Classification Mental Model and Confidence
Collecting confidence data at 3 time points allowed us to track changes in individual participant confidence: 5 of the 8 experienced increased confidence, 1 demonstrated decreased confidence, and 2 maintained similar confidence levels.

The majority of participant responses suggest the characteristics they were using to differentiate harmless and suspicious skin lesions had changed. This indicates participants’ mental models evolved through their interaction with the system. For example, before using the system, participants characterized benign skin lesions primarily by color, shape, and size. At the mid and final interview, participants also included a lack of protrusion as a defining characteristic of a lesion being benign. In discussing malignant skin lesions before accessing the system, comments focused on lesion color, shape, and size. In the mid-way and final interviews, emphasis shifted towards protrusions and protrusion as defining characteristics of malignant skin lesions. This change in perception is linked to participant performance during the testing session, where they scored 7.25 out of 8 on average (Min: 6, Max: 8).

4 DISCUSSION AND CONCLUSION
Feedback about users’ ability to distinguish between normal and abnormal lesions was provided both explicitly through their test scores (at the end of the testing session) and implicitly through the adaptive sequencing that was used to train their visual-perceptual system when errors were made during training. This feedback seems to have supported the development of accurate self-perceptions, as is common when a user is presented with evidence that they may be overconfident (e.g., a low test score when s/he expects a high score or the repetition of items that were answered incorrectly) [11]. It also helps to explain why some users experienced increased confidence while others experienced decreased confidence, which is a pattern that is common in similar types of adaptive training systems that are used to train basic non-visual skills, such as programming and math [12].

In addition to this approach to supporting learning through technology helping users align their confidence with their abilities, MSKBLOCK shaped participants’ mental models. Participants developed their ability to distinguish between observable characteristics of benign and malignant skin lesions by learning the visual indicators of skin lesion class.

Our results imply that future MSKBLOCK system users will increase their ability to properly distinguish between malignant and benign skin lesions. Additional studies should be conducted to further evaluate the mental models developed by users of these types of visual training systems so that we can better understand how their mental models develop and how these might be linked to misconceptions that can be addressed through complementary training approaches. We foresee that the learning techniques denoted through the MSKBLOCK system will have educational applications in not only dermatology, but also in other medical imaging fields that require discrimination in diagnostic imaging such as x-ray radiography, sonography, and magnetic resonance imaging (MRI) technologies.

References
**ABSTRACT**

Japanese foreign language educators encourage proper writing techniques to help students more effectively learn its writing component. Due to written Japanese’s complexity, educational games have tried to make its learning more engaging to reduce frustration. However, existing educational games lack either direct symbol writing input or more sophisticated proper writing assessment. We developed Tensai, a sketch-based educational game that provides an engaging way to practice and receive assessment on proper Japanese writing technique performance. Our game leverages sketch recognition techniques in a Japanese culturally-themed game, and includes time-based objectives to encourage playful and individualized learning of Japanese symbol writing. From our evaluations, we discovered that users generally found Tensai to be engaging and helpful for practicing their Japanese symbol writing.

**Index Terms:** Applied computing—Education—Computer-assisted instruction; Applied computing—Document management and text processing—Online handwriting recognition

1 MOTIVATION

Japanese as a foreign language is one of the most difficult global languages for native English speakers to master, and can take three times longer to achieve equivalent fluency compared to a European language. Written Japanese especially is a labor-intensive endeavor that requires English speakers to expend significant amounts of time, patience, discipline, and perseverance to study its complex and diverse writing script. Educators often introduce stroke writing techniques (e.g., stroke order, direction, and count) for more effectively studying their respective writing scripts. However, valuable personalized feedback of these pedagogical approaches from human language instructors may not be easily accessible outside of classroom hours or for larger classroom sizes.

Continuing advances in computer-assisted language learning tools via mobile apps provide one potential solution to complement existing classroom instruction, by leveraging writing assessment techniques and language gamification for written Japanese. Such tools allow students outside of the classroom to maintain emulated instructor assessment of their Japanese writing technique skill through an entertaining gameplay approach. However, existing educational tools are dominantly constrained by various factors such as lack of writing input (e.g., heavy scaffolding that does not accommodate natural writing), and minimal written technique assessment feedback.

We describe Tensai, an educational application that incorporates automated assessment of students’ proper Japanese writing performance into an engaging sketching game. Tensai adapts sketch recognition techniques for assessment and a Japanese culture-themed visualization to provide Japanese writing practice with real-time feedback. From our evaluations, users felt that Tensai was both engaging and useful for improving their Japanese writing.

Figure 1: Tensai application on laptop and smartphone.

Figure 2: Screenshot of Tensai’s in-game interface.

2 APPROACH

2.1 Expert Interviews

We first recruited two Japanese language instructors—one female—at a large public university in the United States, in order to help guide the design of our proposed educational application. The instructors had at least five years of university teaching experience. From coding the responses of the individual half-hour semi-structured interviews, we discovered the following four key points: 1) writing Japanese correctly for the first time is important, 2) repetitious writing is important, 3) poor spacing in an individual symbol can look separated, and 4) incorrect stroke order and place can make characters look ambiguous.

2.2 User Interface

From the instructor interviews and review of existing educational applications, we created our game’s user interface.

- **Character Health Bar:** Visual indicator of number of available chances for unsuccessful writing attempts.
- **Character Actions:** View of user’s and computer’s black and red ninja characters, respectively. The computer throws a star with prompted symbol to write at the player to destroy or get hit by the star, depending on for writing input correctness.
- **Writing Controls:** Space to write and edit their symbols.

2.3 Recognition System

From the instructor interviews’ third key insight, our recognition system consists of a visual structure and a written technique assessor.

Figure 3: Different stroke visual cues for written technique error: (l) incorrect direction, (c) incorrect order, (r) extraneous stroke.

*e-mails: {sloniger,rupensakariya.c.deguzman.jacobmathews}@tamu.edu
†e-mails: {erwinsusanto22,jungin,ptaele,thammond}@tamu.edu
2.3.1 Visual structure assessment
Our visual structure assessor determines whether the written symbol matches the latest prompted symbol during gameplay, which was adapted from [8].

1. Collect expert’s template data of properly-written symbols.
2. Resample templates and input to 32 points.
3. Scale templates and input into a unit-sized bounding box.
4. Translate templates’ and input’s bounding boxes to the origin.
5. Calculate weighted Hausdorff distance between each template to input.
6. Select template with shortest Hausdorff distance to input. Assign template’s class to input.

Afterwards, the assessor checks whether the selected template’s class matches the class of the user’s input. If there is no match, then we can infer that they are not visually similar.

2.3.2 Written technique assessment
Our written technique assessor addresses the instructor interviews’ fourth key point by partially adapting techniques from [7] to assess incorrect stroke order, direction, and count.

- **Extraneous Count**: Spatially match template strokes to closest input strokes, then re-evaluate matching strokes for visual structure correctness. If visually correct, then extraneous strokes are highlighted. Highlight all strokes otherwise.
- **Insufficient Count**: Spatially match template strokes to closest input strokes, then display unmatched template strokes.
- **Incorrect Order**: Spatially match template strokes to closest input strokes, then temporally compare input stroke order after sorting them by their corresponding template strokes’ time, and finally notify of incorrect order.
- **Incorrect Direction**: For each input stroke spatially matched to unique template stroke, resample input and template stroke to same point count. Perform two calculations of summed point-by-point Euclidean distances between both resampled strokes, both in input’s original and reverse point order. If reverse order’s summed distances are smaller, then notify user of incorrect direction.

3 RESULTS AND DISCUSSION
We recruited 20 university students—5 females—between 19 and 22 years of age, inclusively. Participants self-reported native English and no East Asian language fluency. We introduced to them a limited set of twelve representative symbols from Japanese’s writing scripts (Figure 4).

Participants were introduced to two aspects of our game: game mode and practice mode (an assessment interface with only writing and editing controls). We instructed each participant to perform three writing tasks: 1) continuously write each symbol in practice mode until input counted correct twice, 2) play one session for familiarization, and 3) play one session until game over or voluntary exit. Symbols were presented randomized, and all participants saw familiarization, and no East Asian language fluency. We introduced to them a limited set of twelve representative symbols from Japanese’s writing scripts (Figure 4).

From the responses, we believe that overall reception of the game was very positive. 17 of the participants found the app easy to use, with 15 of those saying they would use it frequently. 15 of the users found the game to be engaging. More importantly, 19 of the 20 participants felt they could improve their handwriting through use of the app. Lastly, 18 of the users considered the character recognition to be fair. We believe that our evaluation demonstrated Tensai’s potential as a character writing educational application for assisting novice students in effectively obtaining valuable feedback on their Japanese script writing performance.

4 CONCLUSION AND FUTURE WORK
In this work, we describe Tensai, an educational game for practicing Japanese writing with intelligent assessment and engaging gameplay. We interviewed two Japanese language instructors for insights on our interface design and features, and developed a sketching game that provides written technique assessment on introductory Japanese writing symbols. From our evaluations, users felt that our system was engaging and has potential for improving their writing.

One potential areas for expanding our work includes providing variable challenges and objectives for users during writing practice, so that they may improve their Japanese writing proficiency. We are also interested in incorporating additional introductory Japanese script symbols for furthering students’ writing practice.

REFERENCES
ABSTRACT

This paper examines the problem of white-balance correction when a scene contains two illuminations. This is a two step process: 1) estimate the two illuminants; and 2) correct the image. Existing methods attempt to estimate a spatially varying illumination map, however, results are error prone and the resulting illumination maps are too low-resolution to be used for proper spatially varying white-balance correction. We show that this problem can be effectively addressed by not attempting to obtain a spatially varying illumination map, but instead by performing illumination estimation on large sub-regions of the image. Our approach (published in [1]) is able to detect when distinct illuminations are present in the image and accurately measure these illuminants. Since our proposed strategy is not suitable for spatially varying image correction, a user study is performed to see if there is a preference for how the image should be corrected when two illuminants are present, but only a global correction can be applied. The user study shows that when the illuminations are distinct, there is a preference for the outdoor illumination to be corrected resulting in warmer final result. We use these collective findings to demonstrate an effective two illuminant estimation scheme that produces corrected images that users prefer.

1 INTRODUCTION

Scene illumination affects the overall color of a captured image. Estimating the illumination and subsequent correction, i.e. white-balance, to remove the color cast caused by the illumination is a fundamental processing step applied to virtually all images. Most white-balance methods assume the imaged scene is uniformly illuminated with a single light source, however, it is not uncommon for a scene to be illuminated by more than one light as shown in Figure 1a. Most existing approaches that attempt to estimate multiple illuminations use a sliding window strategy or image segmentation to perform local illumination estimation. This results in a spatially varying illumination map over the image. Such illumination maps are typically low-resolution (e.g. $15 \times 20$) and their effectiveness in subsequent white-balance correction is often not demonstrated. Moreover, these methods tend to be slow and require prior knowledge that the imaged scene contains two illuminations.

In this paper, we advocate a different strategy for addressing the two illuminant estimation problem. Specifically, we find it more effective not to attempt to estimate a spatially varying illumination map. Instead, we show that applying a single-illuminant estimation method on a relatively small number of large sub-images in the input image can not only detect if two distinct illuminants are present, but provide accurate estimations for these illuminations. To this end, we perform a user study to determine if users have a preference for which illumination they would prefer to be corrected (see Figure 1). Our study found that users have a clear preference for a result that is a mixture of the two illuminations, with more weight on the outdoor illumination.

2 TWO ILLUMINANT ESTIMATION

The overall framework of our method is illustrated in Figure 2b. The image is divided into $4 \times 8$ sub-images. For each sub-image, the multiple regression tree [2] method is applied. Cross-feature consensus is examined on these initial candidates and only candidates in agreement are kept. If the regression tree approach does not obtain a consensus or the collective candidates from the trees have too high variance (set to 0.0001 in chromaticity space in our approach), the results for this sub-image are ignored, otherwise the median of the results is kept as the estimate for that sub-image. Figure 2b shows an example, where rejected sub-images are marked with an $\times$ and those that have passed are marked with a $\checkmark$. After the sub-images have been processed, we are left with a set of 2D illumination estimates in the r-g chromaticity space of the input image. We then compute the pair-wise distance of all candidates. If the average pair-wise distance is less than 0.025, it is assumed there is only a single illumination in the scene and the median of all the candidates is reported as the illumination estimation. Otherwise, the image is classified as having two illuminations, and k-means ($k=2$) clustering is applied and the centroids of the two clusters are taken as the estimates of the two illuminations.

3 USER-PREFERRED IMAGE CORRECTION

We conducted a user study to see if the users would prefer some mixture of the two illuminants to correct the images. The average user choices of each of the 5 illuminant corrections for each category of images are shown in Figure 4b along with their 95% confidence intervals represented by error bars. From this result, we see that Cat. I (two distinct illuminations) have a clear preference leaning towards the correction using a higher weight of the outdoor illuminant (i.e. the $L_3 = 0.25L_1 + 0.75L_2$ illuminant). For Cat. II (two similar illuminations) the preference is less pronounced and slightly favors an average result. This is consistent with the finding in [4] that visual difference illuminant corrections within $3^\circ$ is not noticeable.

4 RESULTS

Figure 3 shows some examples for two-illumination images. To have a comparison, the Corrected Moments [3] method was used.
Multiple regression trees illuminant estimation

- \( f^1 \): average color
- \( f^2 \): brightest color
- \( f^3 \): dominant color
- \( f^4 \): chromaticity mode

(a)

Figure 2: (a) This image provides an illustration of the regression trees method proposed by Cheng et al. [2]. This method produces a set of candidate estimates in the 2D rg-chromaticity space. The median of the candidates is used as the final estimate. (b) This image is an overview of our two-illumination method. The image is divided into sub-images. The method in (a) is applied on each sub-image. If the illumination estimate candidates obtained by (a) per sub-image are similar, the estimate result is kept (denoted with a \( \checkmark \), otherwise they are rejected denoted with an \( \times \)). The final set of reliable estimations (i.e. those kept) are examined to see if they form one or two clusters, which are used as the illuminant estimates.

(b)

Figure 3: Visual comparison of image global correction. The three images are from the Gehler-Shi data set [5, 6]. The Corrected moment [3] result is compared.

to represent single illuminant estimation methods. We can see that for these images, the Corrected Moments method tends to give the indoor illuminant estimation or the mixture of indoor/outdoor. These illuminant estimates make the corrected images bluish in nature. In contrast, our results are closer to the user preferred correction.

REFERENCES

Figure 4: (a) User preferences for the indoor \( L_1 \) and outdoor illuminant \( L_5 \) corrections. (b) User preferences for each of the 5 illuminants, over the two categories (distinct and similar illuminants). Error bars represent the 95% confidence intervals.
Revisiting Autofocus for Smartphone Cameras

Abdullah Abuolaim
Lassonde School of Engineering, York University, Canada

ABSTRACT

Autofocus (AF) on smartphones is the process of determining how to move a camera’s lens such that certain scene content is in focus. The underlying algorithms used by AF systems, such as contrast detection and phase differencing, are well established. However, determining a high-level objective regarding how to best focus a particular scene is less clear. This is evident in part by the fact that different smartphone cameras employ different AF criteria; for example, some attempt to keep items in the focus, others give priority to faces while others maximize the sharpness of the entire scene. The fact that different objectives exist raises the research question of whether there is a preferred objective. This becomes more interesting when AF is applied to videos of dynamics scenes. Our work in process aims to revisit AF for smartphones within the context of temporal images. As part of this effort, we describe the capture of a new 4D dataset that provides access to a full focal stack at each time instance in a temporal sequence. Based on this dataset, we have developed a platform and associated application programming interface (API) that mimics real AF systems, restricting lens motion within the constraints of a dynamic environment and frame capture. Using our platform we evaluated several high-level focusing objectives and found interesting insight into what users prefer. We believe our new temporal focal stack dataset, AF platform, and initial user-study findings will be useful in advancing AF research.

Index Terms: autofocus, focal stack, AF platform, computer vision

1 INTRODUCTION

One of the crucial steps in image capture is determining what part of the scene to focus on. In this poster paper, we examine this problem for smartphone cameras because smartphones now represent the dominant modality of video and image capture performed by consumers. The goal of AF is straightforward. Given some high-level objective of what scene content or image region is desired to be in focus, AF systems attempt to move the lens such that these regions appear sharpest. The low-level algorithms used to determine image sharpness—for example, contrast detection and phase differencing—are well established. What is more challenging is using these low-level algorithms to realize high-level AF objectives for dynamic scene content in a temporal image sequence (i.e., video). This is evident from the variety of different AF criteria used by different smartphone cameras. Figure 1 shows an illustrative example. In this example, an Apple iPhone 7 and a Google Pixel have captured a scene with objects that move on a translating stage. The translating stage and controlled environment allows each camera to image the same dynamic scene content. We can see that each camera is focusing on different image regions at the same time slots in the video.

This begs the question of which of these two approaches is preferred by a user. From a research point of view, one of the major challenges when developing AF algorithms is the inability to examine the full solution space since only a fixed focal position can be captured at each time instance. While it is possible to capture a full focal stack for a static scene, it is currently not possible for a temporal image sequence in a dynamic environment. Moreover, there are additional constraints in an AF system beyond determining the right focal position given a full focal stack. This lack of access to (i) temporal focal stack data and (ii) an AF platform that holistically incorporates lens motion, scene dynamics, and frame advancement is the impetus for our work.

Contribution The contribution of this work is a software platform for AF research and an associated 4D temporal focal stack dataset. Our AF platform allows the design, testing, and comparison of AF algorithms in a reproducible manner. Our focal stack dataset is composed of 33,000 full-frame images consisting of 10 temporal image sequences, each containing 50-90 full focal stacks. Our software platform provides an AF application programming interface (API) that mimics the real-time constraints, including lens motion timing with respect to scene motion and frame advancement. Additionally, we have performed analysis on several smartphone AF algorithms to come up with a set of representative high-level AF objectives. Using our platform and data we have implemented these algorithms to produce similar outputs found on real phones and used the results to perform a user study to see if there are any preferences. Our user study reveals that overall lens motion, and not necessarily the actual scene content in focus, is the predominant factor dictating preference.

2 RELATED WORK

Focal stack datasets Beyond various ad hoc focal stack data available online from class projects and photography enthusiasts, there are very few formal focal stack datasets available for academic research. Two notable datasets are by Mousnier et al. [6] and Li et al. [5]. The dataset in [6] provides 30 focal stacks of static scenes of images of size 1088 × 1088 pixels. The dataset in [5] captured 100 focal stacks of image size 1080 × 1080 pixels, again of static scenes. The number of images per focal stack ranges from 5 to 12. These datasets are not intended for the purpose of AF research, but instead target tangentially related topics, such as digital refocusing [2,3,10], depth from defocus [1,9], and depth from focal stacks [8]. The focal stacks in these datasets are synthetically generated based on the Lytro light field camera [4,7]. Unfortunately, the consumer-level Lytro devices do not support video capture. The new Lytro Cinema does offer video light field capture, but the cost of renting this device is prohibitively high (in the hundreds of thousands of dollars). Moreover, the Lytro Cinema is not representative of smartphones. Unlike the datasets in [5,6], our dataset provides a much larger focal stack of 50 images of size 3264 × 1836 pixels, and consists of 10 temporal image sequences with up to 90 full focal stacks per sequence.
3 AF ANALYSIS AND DATASET CAPTURE

We analyzed the performance of three representative consumer smartphones (Apple iPhone 7, Google Pixel, Samsung Galaxy S6) to observe their behavior under different scenarios. The cameras are positioned such that their fields of view are as similar as possible. The frame rate for video capture is fixed at 30 frames/sec. We experimented with a wide range of scene configurations, such as an object with a figurine with a human face, textured backgrounds, and various moving objects. As previously illustrated in Figure 1, we observed that the AF behaviors differ between phones.

Based on our observations, we settled on 10 representative scenes that are categorized into three types: (1) scenes containing no face (NF), (2) scenes with a face in the foreground (FF), and (3) scenes with faces in the background (FB). For each of these scenes, we allowed different arrangements in terms of textured backgrounds, if the camera moves or not, and how many types of objects in the scene change their directions (referred to as motion switches). Due to page limits, a table of scene settings details will be presented during the poster session.

For each of these 10 scenes, we captured the following data. First, each scene was imaged with the three smartphone cameras. This video capture helps to establish high-level AF objectives used on phones and determines the approximate video length needed to capture the overall scene dynamics. Next, we captured temporal focal stacks for each of these scenes. We refer to these as image sequences to distinguish them from the actual videos. To capture each image sequence, we replicated the video capture in a stop-motion manner. Specifically, the objects in the scene are moved in motion increments of 3.87mm between consecutive time points. We used the Samsung Galaxy S6 Edge to perform the image capture using a custom Android app that fixed all camera settings (e.g., ISO, white balance, shutter speed). Our app also controlled the lens position, such that for each time point $t$, we captured a focal stack of 50 images where the camera lens is moved in linear steps from its minimum to maximum position.

Figure 2 shows an example of scene 2 with 50 time points. Each time point $t_i$ in Figure 2 has a focal stack of 50 images that are denoted as $I_i^j$, $j = 1,...,50$, where $i$ denotes time point and $j$ indexes the focal stack image associated to a specific lens position.

4 AF PLATFORM AND API

The CDAF and PDAF process can be divided into three main steps: first, determine a desired region of interest (ROI) based on the high-level AF objective; second, measure the sharpness or phase of the ROI selected; third, adjust the lens position to maximize the focus.

Based on the observed behavior of the captured video from our three smartphone cameras on the 10 scenes, we determine four high-level AF objectives in terms of ROI as follows: (1) global ROI targeting the whole image; (2) a layout of 9 focus points with 9 ROIs; (3) a layout of 51 focus points with 51 ROIs (similar to the global ROI); (4) and a face region ROI where the largest region of detected faces is set as the ROI.

Our AF API is designed to emulate AF in smartphones. The platform and API imposes constraints on lens motion timing with respect to scene motion and video frame rate. As such, our API and platform have a local and global virtual clock. The local clock, denoted as $C_{loc}$, emulates the real-time internal clock on the smartphone, whereas the global clock, $C_{glob}$, emulates the real-world timing (scene dynamics).

Platform Timing Since the Samsung Galaxy S6 was used to capture our dataset, we measured its performance to establish the mapping between the local and global clocks. The Samsung Galaxy S6 Edge requires 42 ms to move the lens one step (including image capturing and AF processing). The time required for the translating stage motor to move one step (3.87mm) is 414 ms. Recall that a single translating stage motor step in real time is equivalent to a discrete time point in our stop-motion setup. Therefore, the number of steps $s$ allowed for the lens to move in one time point is equal to $414/42 \approx 9.86$ steps. Based on this approximate calculation, we fix $s$ to 10 steps and we relate $s$ to the local clock $C_{loc}$ (one [1] lens movement costs one [1] clock cycle). Accordingly, the corresponding global clock $C_{glob}$ increments every 10 clock cycles. Due to page limits, a detailed table of our API calls will be presented during the poster session.

5 USER STUDY ON AF PREFERENCE

We conducted a user study to determine if there was any particular preference for the different the AF methods. We observed that for NF videos, the global (GB) AF objective is the most preferred. For the FF videos, the face region (FR) AF objective is the most preferred. For the FB videos, there is no strong preference among the three objectives GB, 51 focus points (51 FP), and FR, but the most preferred is GB followed by FR. Additionally, we calculated the 95% confidence intervals for these results as represented by the error bars, which indicate the statistical significance of the results. We observed that there is a clear correlation between user preference and lens movements, suggesting that users tend to prefer the objectives with fewer lens movements. This is indicated by the negative correlation coefficients shown on the plots. For the second category that contains a prominent face in the foreground, the results suggest that users prefer the face AF that locks onto the face even if more motion of the lens is required to achieve this objective. Due to the page limit, a table of our API calls details will be presented during the poster session. Due to the page limit, a detailed discussion and figures will be presented during the poster session.

REFERENCES