Human Computation in Visualization: Using Purpose Driven Games for Robust Evaluation of Visualization Algorithms

Nafees Ahmed, Ziyi Zheng and Klaus Mueller, *Senior Member*, *IEEE*

Abstract—Due to the inherent characteristics of the visualization process, most of the problems in this field have strong ties with human cognition and perception. This makes the human brain and sensory system the only appropriate platform for evaluating and fine-tuning a new visualization method or paradigm. However, getting humans to volunteer for these purposes has always been a significant obstacle, and thus this phase of the development process has traditionally formed a bottleneck, slowing down progress in visualization research. We propose to take advantage of the newly emerging field of Human Computation (HC) to overcome these challenges. HC promotes the idea that rather than considering humans as users of the computational system, they can be made part of a hybrid computational loop consisting of traditional computation resources and the human brain and sensory system. This approach is particularly successful in cases where part of the computational problem is considered intractable using known computer algorithms but is trivial to common sense human knowledge. In this paper, we focus on HC from the perspective of solving visualization problems and also outline a framework by which humans can be easily seduced to volunteer their HC resources. We introduce a purpose-driven game entitled "Disguise" which serves as a prototypical example for how the evaluation of visualization algorithms can be mapped into a fun and addictive activity, allowing this task to be accomplished in an extensive yet cost effective way. Finally, we sketch out a framework that transcends from the pure evaluation of existing visualization methods to the design of new ones.

Index Terms— Human Computation, perception, evaluation, color blending.

1 Introduction

"A picture is worth a thousand words". The human brain and its visual sensory system are constructed in such a way that they work like a broad pathway between the information and the human mind. The process of visualization capitalizes on this well-known idea and provides a way of having deeper insight into the data by transforming them into a visual representation. Success of such methods thus depends greatly on how well they are able to convey the message to the viewer. Over the years we have seen many different techniques and methods that try to conform to the knowledge we have about human perception to derive better visualization algorithms [32, 35]. Due to the limitation of silicon-based computers and also the fact that we are still far away from actually building a complete analytical model of the human brain's working process, truly optimizing visualization to capture its full potential seems hard to achieve. But, the recent growth in the field of human computation has opened up new ways to look at the problem from a completely different perspective. The possibility that the human brain and sensory system can be brought into the loop for computation provides an opportunity of revisiting the entire process of design, evaluation and generation of visualization.

In a sense, computing with humans is an age old concept. Before modern day computers were invented to ease the pressure on mankind, all sorts of computations were done by what we may call "human computers" [20]. But, the human brain was never very good at crunching numbers in a fast and accurate manner. So, computing machines were invented and their power is growing every day. For long, we have been only a user to such systems and the idea has grown on us so much that a simple truth gets often overlooked – silicon-based computers are definitely extremely fast, but they are only good at a limited type of problems. There are many functions that are much better suited for human brains to solve, even though brains have much slower compute cycles. So, ideally one would want to solve a problem on a platform that is best suited for its type. This brings about the scenario of hybrid computational platforms comprised of CPU, GPU and Human brain (HPU) [18], which can achieve success in solving a much wider variety of problems than silicon-only systems. Of course this will require a framework that can efficiently distribute and organize the tasks among the human computers. Fortunately, with the advancement in internet technologies and services specifically targeting crowd-sourcing, bringing in the collaborative power of the global population and building a working systems is quite feasible.

In this paper, we discuss in detail how the field of visualization can greatly benefit from bringing humans into the computational loop. We revisit the pipeline for visualization from a human computation point of view and look for possible improvements in different stages of the pipeline. One very strong and immediate candidate for improvement is the process of evaluation in visualization. Quantititative evaluation of visualizations has always been a challenge, because in most cases a well accepted metric is hard to find, leaving user evaluation as the only accepted method for this task. Generally in visualization, the problem parameter space is tremendous, requiring an unrealistically large number of datapoints to be collected for a comprehensive evaluation. Arranging that many people is hard, costly and time consuming. To illustrate how human computation can help solve such evaluation problems, we present a purpose-driven game called "Disguise". As an example, we consider the problem of evaluating color blending algorithms with respect to their performance to enable transparency perception. We do so by formulating the evaluation function into a game-driven human algorithm. With the help of this game, we show how we can collect a sufficient number of data points to perform a detailed evaluation of 4 blending algorithms. We show that data collection in such manner is faster, cheaper than other existing methods and also comparably reliable. We also discuss on how to build on such a concept and extend the process into the design phase of a visualization.

Our paper is structured as follows. Section 2 presents related work. Section 3 provides background on the concepts of Human Computation. Section 4 revisits the visualization pipeline and discusses in general how we can benefit from having HPU based computation in it. Section 5 gives an illustrative example of game based human computation algorithm, from design to evaluation. Section 6 discusses how to extend the same method into other steps of the pipeline and gives pointers to possible future research directions. Section 7 ends with conclusion.

2 Related Work

Even though the world of visualization is yet to take advantage of having humans as a computational resource, the concept itself is not particularly new. In the past two decades, the growth of the internet has opened up the opportunity of bringing the global population into

- *Nafees Ahmed, Ziyi Zheng and Klaus Mueller are with the Visual Analytics and Imaging Laboratory, Computer Science Department, Stony Brook University, NY. Email: {nuahmed, zizhen, mueller}@cs.sunysb.edu*

a common platform for large scale collaboration. People who realized the potential of such efforts created systems that can be considered to be first true step towards social computing. One example of such work is "The Open Mind Initiative" [44], which was dependent largely on volunteer-provided data for constructing better software solutions. In 2004, Luis von Ahn came up with a purpose driven game called "ESP" [7] that utilized human observation for labelling digital images, showing the power of computing with humans in solving an important problem in computer vision. This was followed by a series of similar works [8-10, 30], which eventually led to the formalization of the term Human Computation (HC) [29]. HC started off with purpose-driven games, but with the introduction of micro task-based crowd-sourcing platforms like Amazon Mechanical Turk (MTurk), integrating the human processor into the flow of an actual computational process became feasible. "VizWiz"[12] is a great example of such a system. It gives a near-real time answer to any question related to a picture taken on a cell-phone by immediately creating a task on MTurk. "Soylent" [11] is another clever application where computationally hard word-processing functions are handled by MTurk workers. Such systems inspired researchers to consider the HPU (Human Processing Unit) [18] as an integral part of the computing architecture and in the past two years we have seen its application in many different fields of computer science [19, 23, 28, 33]. In all these examples, the problems considered were generally simple in formulation and the games, in most cases had the advantage of finding simple mapping between problem statement and game parameters. In 2010, another multiplayer game titled "fold.it"[16] showed how even very complex scientific problems can be formulated as a multiplayer online game. Such promising results of HC inspired us to tap its power into visualization field.

In visualization, the concept of bringing crowds into the loop of computation is a relatively new trend. With the recent success of cost effective crowd-sourcing through MTurk, we have seen studies done on the viability of using MTurk for large scale user studies [15, 22, 25, 26]. These works provide in-depth discussions about pros and cons of crowdsourcing as a platform for doing user studies but their analysis is only limited to that scope. As such, these efforts can only be considered as crowd-sourced user studies rather than computing with humans. Hence, we are yet to see a true adoption of HC based techniques to visualization research. This makes the contributions from this paper the first of its kind in this field.

Our example system, "Disguise" shows a human computation way of evaluating visualization algorithms. Evaluating a visualization technique has always been considered to be a challenging task [36]. With the lack of quantifiable intrinsic quality measures [13], the only acceptable solution towards measuring success of an algorithm is to do a user evaluation. In this paper, we evaluate transparency perception of color blending algorithms. Of the algorithms we take into consideration [14][31][33][37], only one of them [32] has provided a detailed user evaluation. Even then, due to the fact that arranging people to do a user study is always a time consuming, costly and difficult job, the scale is rather limited to an order of a thousand data points. Compared to that, our method provides the opportunity of performing evaluations based on millions of data points, with comparable reliability, speed and lower cost.

3 Human Computation: Concepts and Background

In simple words, "Computation" is the process of transforming input into a desired output following a predefined procedure or algorithm. Human computation thus means a method of computing by assigning partial/complete tasks to humans. To illustrate the idea clearly, consider the algorithm **quicksort**. As input **quicksort** needs two items – 1) An array of objects to be sorted and 2) A function that can compare two elements in the given array. We can use this algorithm to sort an array of any type of object as long as we have a valid compare function. There are known fast algorithms that can compare two numbers or even some other complex programming constructs quickly. But, what if we asked this algorithm to sort an array of images taken from a surveillance camera based on how suspicious the activities are. Even with the vast modern day advancements in the fields of Artificial Intelligence and Computer Vision, this comparison function between two images is nowhere near perfect. But, the human brain can comprehend the content of an image very quickly and easily. So, we can replace the existing **compare** function with a call to the human processor, **human_compare.** As long as this function returns a valid comparison, we have a sorting function that will baffle a great many of artificial intelligence algorithms by performance. The function **human_compare** is a call to the human computer, which means a person will judge the two pictures and compare them. One possible way of implementing the function is to programmatically produce micro tasks in a crowd sourcing platform like Amazon Mechanical Turk using an API like TurKit[31].

There are three common techniques used to compute algorithms using humans: use of crowd-sourcing, purpose driven games and recruiting people by providing mutually beneficial service.

Crowd-sourcing: Presence of a strong web-service driven SDK from services like Amazon Mechanical Turk allows a programmer to view the workers on MTurk as some computational resource that can magically execute complex human functions with ease. As discussed earlier, examples of such techniques are [11, 12]. The advantage of using this technique is the simplicity of mapping the problem instance into an executable program. But, this incurs a monetary cost on computational power because of the remuneration of the workers.

Purpose driven games: In this approach, the problem instance is mapped into an entertaining gaming activity. Players play these games for fun and help compute the function for free. An early example system of this kind is the "ESP" [7] game. This game mapped the computer vision problem of producing image labels into a fun multiplayer gaming activity. The advantage of creating such a game is that it can compute the function with considerable amounts of reliability and volume, and at very minimal runtime cost. But, creating such games is not easy and making a game popular enough to produce useful amounts of data is more of a social engineering task than exact science.

Recruitment by mutually beneficial service: These systems are somewhat similar to purpose driven games, but rather than providing entertainment toward the users, it provides service value. A famous example system of such kind is reCaptcha [6]. In this system, people type in deformed scanned words to prove that they are indeed human. In the meantime, they help digitize books and thus working as a human OCR system. Just like purpose-driven games, they provide very a high productivity/cost ratio, but are harder to design.

In this paper, we show a design example of a purpose driven game to solve visualization problems. According to [29], the design of a good human algorithm-based game should have the following:

Function Mapping: The player is presented with a challenge (the problem), an objective function that he/she is required to optimize (score, experience etc.) and a set of actions to choose from. The challenge has to be designed in such a way that it maps the target function task indirectly. That is, we should not simply ask the player to provide their perceived ordering of the layers. Otherwise it becomes a crowd-sourced task, not a game.

Game Feedback: Performing any of the available actions changes the state of the game. Each such action can produce one of the following three outcomes – positive, negative or neutral. In an ideal gaming scenario, the player only tries to optimize his/her objective function. Hence, the player will prefer positive actions over negative ones and occasionally choose neutral ones for strategic purposes. We can control the behaviour of the user by carefully designing the actions and their outcomes. We design the set of actions and outcomes to ensure that actions selected by the player helps compute our function.

Entertainment Value: The game has to be fun to play with. To ensure maximum throughput, it should ensure features that brings people back. But, at the same time, these features should not by any means affect the computation.

Exclusive to humans: A true human computation game should not be playable by a programmed bot. Otherwise, a computer can compute the function by playing the game by itself and thus nullifying the whole need of having a human driven algorithm.

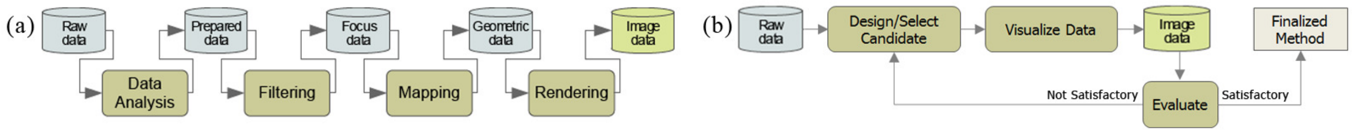


Figure 1. (a) Visualization pipeline [4]. (b) Visualization design workflow.

4 Visualization with Human Computations

We discuss the possible inclusion of human based computation in visualization from two different perspectives: The process of rendering a visual representation and the process of designing a new visualization scheme.

4.1 Rendering a visualization

Figure 1(a) shows the steps in the most accepted view of the visualization pipeline [4] as inspired by [21]. We make a quick review of the steps involved with respect to Human Computation.

Data Analysis: Raw data generally require processing before they can be ready for visualization. This involves a set of computer centric methods, such as the application of smoothing filters, interpolation for missing data, corrections for erroneous data, removal of noise etc. In the case of scientific visualization, this phase generally involves handling huge amounts of numeric data, which is better suited for computers. Information visualization on the other hand, might benefit from human computers in cases where the data source itself is too abstract to be handled by computers directly.

Filtering: Filtering involves selection of the data to be visualized. This phase is mostly user centric and is heavily dependent on a suitable user interaction tool. In a sense, this phase is already being processed with the help of the human brain.

Mapping: Once the target data points are identified, this step maps the focus data to geometric primitives (e.g. points, lines) and their attributes (e.g. color, position, size). Mapping is considered to be the most critical in terms of achieving success in visualization and contains many open problems in this research field. The transformation from data points to geometric interpretation is driven by a set of parameters (e.g. transfer function). Producing an effective and expressive image representation requires the system to find an optimal set of values for such parameters. Thus, the computation in this phase can be thought of as an optimization problem over the whole parameter space to find the best possible image representation. The commonly practiced method either involves computer guided interactive tools [17, 46] or automated computer driven optimization [35, 40]. We argue that such optimizations generally involve steps that are better suited for the human brain. In Section 8 we discuss some of the possible ways of achieving such feat.

Rendering: Rendering of an image from geometric data is predominantly a heavy numerical process best suited for computers.

We note that the process of visual analytics puts this pipeline into an iterative loop where the analysis etc. is driven by some need specified by the user who becomes part of this loop.

4.2 Designing a visualization

Designing a new visualization is more of an open, creative and an on-demand process. Trying to find a common framework for such process is challenging. But, we can think of the workflow in such case as an iterative process looping between design and evaluation, as shown in Figure 1(b) [45]. In this figure the block "Visualize Data" contains the Analysis, Filtering, Mapping, and Rendering steps of Figure 2(a).

Design: In necessity, this step covers almost anything related to coming up with a process of visualization once we have data in our hand. It is generally handled by an expert in visualization with domain knowledge in the data. It starts with deciding the type of visualization that might be applicable for the given data. Then, following the pipeline shown in Figure 1(a), a sample set of images are generated which need to be evaluated to see how good the technique was. The design phase can be aided by human computers if the process of visualization itself can benefit from having human in the loop. (One possible example is given in Section 6.)

Evaluate: Any design of a visualization method requires evaluation for verification. This evaluation step either quantitatively or qualitatively judges if the method under consideration produces satisfactory performance. Depending on the result of the evaluation, the method is either revised or finalized. A good evaluation method is considered crucial in designing a good visualization scheme. As discussed earlier, visualization being a perception based process, the only true evaluator of a produced image is the human. We argue that the process of evaluation can be improved greatly by considering human computation based methods. Since this part of the visualization design has the most obvious connection with the human; we present a detailed example of such an evaluation method.

5 Disguise: A Game to Evaluate Algorithms

As an example of how human computation can be useful in evaluating algorithms in visualization, we created a purpose-driven game entitled "Disguise". The game helps evaluate the performance of four algorithms designed to compute the color blending between semi-transparent layers for perception of layer ordering. We devote this section to discuss how to design the game as an implementation of a human computation algorithm.

5.1 Problem statement

First, we need to define what function we intend to compute using humans. We begin by exactly defining the transparency perception in color blending. Let us assume, there are two semi-transparent layers on top of each other. Each of the layers has a particular color and an alpha value associated with it to represent its transparency. Hence, their color state can be represented by an RGBA quadruple. Let layer A and layer B be represented by the quadruples $A_{RGB\alpha} = (A_R, A_G, A_B, A_\alpha)$ and $B_{RGB\alpha} = (B_R, B_G, B_B, B_\alpha)$ respectively. Also, it is given that layer A is on top of layer B. Then, the blending between these two layers would be a composite color represented by the following function,

$$C_{RGB\alpha} = \textbf{blend}\,(A_{RGB\alpha}, B_{RGB\alpha})$$

Figure 2 shows an example of blending between two semi-transparent layers. Layer A and B has the color Green and Blue respectively with an alpha value of 0.7. The composite color $C_{RGB\alpha}$ is produced by following the Porter and Duff over operator [37]. Among many other things, an important desired feature in such color blending is the ability to perceive the ordering of the layer from the composite color. This property is defined as perceptual transparency [34]. Transparency perception is affected by many different aspects, including luminance, chromaticity, parallax motion, stereo depth,

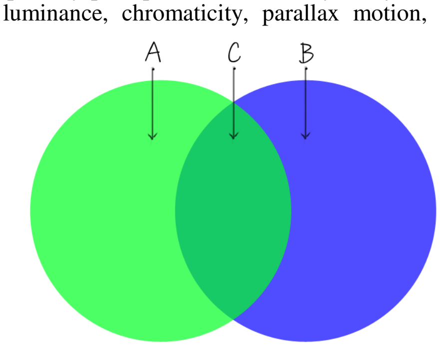


Figure 2. Color blending and transparency perception. Composite color C is produced from the blending between A (green) and B (blue). The human eye perceives A to be on top of B.

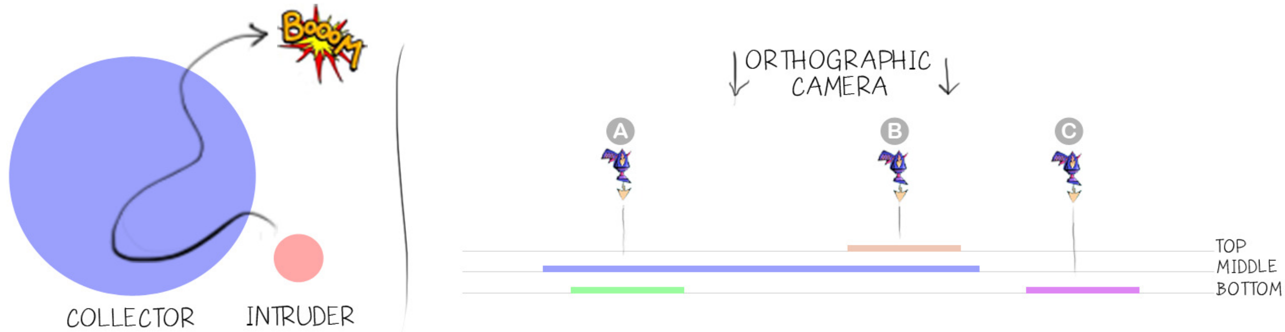


Figure 3. (Left) Illustration of how an intruder moves randomly over a collector before self-explosion. (Right) A side-view of the game environment showing the three layers. The collectors are always in the middle layer while intruders can float around either on the top or the bottom layer. The markers A, B, C show three possible action scenarios from the player. Case A: The intruder is a layer below the collector. Striking here hits the collector and damages it. Case B: The intruder is a layer above the collector. Striking here disables the intruder keeping the collector intact. Also, since it is on top of the collector, the player scores a point. Case C: The intruder is in either layer but outside the collector radius. Striking here disables the intruder but no points are scored or damage is done to the collector.

organization of figures, subjective contours etc. Here, we put aside every property other than low level per pixel composition of color. We want to evaluate the performance of a given blending algorithm by its ability of exposing transparency perception when the only information we have is the perceived color of the individual layers and their composition. Hence, the function we intend to compute with a human processor is,

$$S = \text{evaluate\_blend}\ (b)$$

Where,

$b =$ a blending algorithm
$S =$ performance of algorithm b in transparency perception

In this example, we evaluate the following algorithms,

Algorithm 1 The Porter and Duff over operator [37]. It is the most well-known algorithm for compositing two semi-transparent color. The blend function in this respect is defined as,

$$C_{RGB\alpha} = A_{RGB\alpha} \oplus B_{RGB\alpha} = \alpha_A A_{RGB\alpha} + (1 - \alpha_A)\alpha_B B_{RGB\alpha}$$

Algorithm 2 Hue preserving color blending [14]. In this algorithm, to avoid the generation of false colors, the composite between front and back color is confined within either of these two hues.

Algorithm 3 Local color blending method [32]**.** In this technique, the color of the background is de-saturated by a constant factor keeping its lightness constant. The new color is then blended with the foreground using algorithm 1.

$$B'_{RGB\alpha} = \text{de-saturate}\ (B_{RGB\alpha})$$
$$C_{RGB\alpha} = \text{local\_blend}(\mathrm{A_{RGB\alpha}}, \mathrm{B_{RGB\alpha}}) = A_{RGB\alpha} \oplus B'_{RGB\alpha}$$

Algorithm 4 A variation of Algorithm 3, with the back layer blurred using a Gaussian filter before passing it to the blending process. We included this algorithm for a very particular reason. The perception of transparency is governed by several factors. One is due to the so-called episcotister model [34] which advocates that luminance differences play a major role. Another factor is the Michelson contrast [42] which states that the contrast of detail on a surface is lowered when a semi-transparent layer superimposes it. Finally, the blurring of detail on the back surface is another perceptual clue [41], but one which does not necessarily reduce the Michelson contrast. Hence, this algorithm provides a strong visual cue or layer ordering on top of mere color blending. We added this algorithm to the list to verify that our game can actually evaluate transparency perception truthfully.

5.2 Algorithm design

Let us first consider a simplified case of **evaluate_blend,** which asks us to evaluate a blending algorithm given a pair of colors as foreground and background.

$$s = \textbf{evaluate\_blend\_simple}\ (b, f_{RGB\alpha}, b_{RGB\alpha})$$

Where,

$b =$ a blending algorithm
$f_{RGB\alpha}$ = foreground color
$b_{RGB\alpha}$ = background color
s = can the human eye perceive color $f_{RGB\alpha}$ on top of color $b_{RGB\alpha}$ when compositing is done using algorithm b?

Crucial to our work is the question if we can compute this function using present day computers. The answer is of course no, because this would require an algorithm that can simulate human perception. However, we can always create a very simple test just like Figure 2 and ask a human to judge which layer he/she perceives to be on top. We can replicate the test for many people and judging by their correctness in response, we can assign a score.

A complete evaluation of our original objective function can now be done if we can compute **evaluate_blend_simple** for all possible foreground and background color pairs. The total number of combinations generated in this way is a massive number and might seem to be overkill, but for simplicity and correctness, for now we would like to define it like the following. Let us assume, we can actually arrange for this huge number of human computational resources to gather scores over the whole color spectrum. Once we have this, we can then compute the function **evaluate_blend** on any computational platform of our choice, given an evaluation metric that depends on the scores received. In our implementation, rather than finding all possible foreground and background combinations, we choose a stochastic approach. We select a foreground and background color by randomly picking their RGBA component values from a uniform distribution over 0.0~1.0. We base our computation on the data received and provide an approximate computation of the function **evaluate_blend**. We argue that as we make more successful calls towards the function **evaluate_blend_simple**, we increase our accuracy in approximating **evaluate_blend.**

5.3 Game Design

5.3.1 A general description of the gameplay

"Disguise" is made following the footsteps of single player classic arcade games that duel on the player's skill of perception and reflex.

The game introduces a story about a distant planet, which is sadly under attack from notorious disc shaped "Intruders". Intruders are small semi-transparent circles with a single color. They fly into the screen, move around randomly and then after some time, blow up. When they blow up, they cause some damage. To survive, the player needs to disable them. To help the player catch these intruders, the game provides five defense elements, called "Collectors". Collectors are larger circles that stay still, distributed randomly over the playing area. They are also single colored and are semi-transparent. Intruders and collectors are arranged in layers. All the collectors are placed in the middle layer. The intruders, on the other hand can be either a layer above or below the collectors. Since both collectors and intruders are semi-transparent, the player always perceives a blended view of discs moving around. This in some sense acts like a camouflage for the intruders, and hence the title of the game "Disguise".

Now, the player, being the saviour of that planet, is endowed with a powerful weapon. Left click on the screen fires the weapon and it can blast off both intruders and collectors. To disable an intruder, the player has to aim and shoot at them. If the player can successfully disable an intruder while they are on top of a collector, the broken pieces from the intruder falls into the collector and the planet gathers valuable information about the invasion and the enemy. This way the player scores points. On the other hand, if the player misjudges the intruder to be on the top-layer when they are actually a layer below the collector, the strike from the mighty weapon hits the collector and damages it. The collectors can only take a limited amount of damage before they break down. So, in a nutshell, the player has the job to protect the collectors (and thus the planet) by identifying the intruders that appear to be on the top layer and disabling them, before either the explosions or the misjudgements kills all the collectors. The game play basics are illustrated in Figure 3.

5.3.2 Game feature details

In this section, we describe both game design and rules detail. Arcade games are fun and addicting. This generally comes from two aspects of the game – the challenge of learning a skill and the sense of progression through the game. Different aspects of the gameplay rules and their design motivations are explained below.

Intruder Entrance Interval: After the game starts, a new intruder spawns at defined intervals. Also, every time an intruder is disabled, a new one is introduced. Thus, the pace of the game adapts a little with the skill of the player.

Intruder self-destruction: Each intruder is assigned a lifetime, after which they can either dissolve or explode. Since we want our players to guess only the discs that are on top, we only allow the intruders on the top layer to explode. If an intruder explodes on top of a collector, the collector takes some damage. Also, as a double penalty the score is reduced a little. So, the player is pushed to judge well which intruders are harmless (from bottom layer) and which are going to cause trouble (from top layer). This indirectly gives a negative feedback to players who refrain from giving any input.

Score Multipliers: Every collector has a score multiplier associated with it. Each time the player makes one correct guess, the multiplier is increased by one. The next correct guess on the same collector will thus bring a lot more points. A misjudgement or an explosion on top of collectors will reset the multiplier. This particular feature ensures that a person, who is playing carefully and judging well, will score much more quickly than one who is reluctant to distinguish between top and bottom layers.

Leveling Up: We provide a sense of progression to the players by dividing the game into multiple levels. The game starts at level 1. Each of the levels has a predefined barrier of score. The player goes into the next level once he goes beyond that score. The level of the game indicates how hard it is to play. Several aspects of the game are functions of the current level. The first level-dependent feature is the interval of intruder entrance and speed. As the player levels up, the intruders come in at a faster rate and move around at a faster pace, challenging him/her to make decisions more quickly. The second such feature is the transparency of the game elements. The alpha values of the intruders and the collectors are chosen from a distribution that is a function of the current game level. This ensures that at early stages of the game, the player is given a task that is easier to achieve. But later on, once he/she is well tuned and well trained, the probability of getting some hard to perceive blends will increase. When the player levels up, he/she is given a set of five new collectors, resetting any damages they accumulated in the previous level.

Feedbacks: We faced one big challenge in making the game. Because we needed to maintain a controlled experimental design, we had to keep the character look and feel limited only to very simple looking circles with uniform colors. Players want a game that looks nice and feels nice. But, we could not use textures, perspective camera, lighting or any other visual effect that has an impact on color perception. So, we decided to bring a few features that would make the characters more realistic but would not disrupt our experiment. First, we added a physics driven mass spring model to the collectors. So that every time they are hit by the weapon, they will wobble, giving users a visual feedback of the situation. Second, we added some comic effects that are only introduced when there is an explosion or disappearance. Third, we introduced sound effects. This is particularly important in keeping the players entertained without interrupting visual perception.

Shuffle: Consider a scenario when the player only has one or two collectors remaining and they only cover a very little part of the screen compared to the gaming area. This situation might make the game a little boring because the intruders might not always go through that region. To ensure that the player has a way around this, we added a shuffle button that takes the current set of collectors and rearranges them in the screen. It also introduces a new color for them. So, if the player is not happy with their collector color or their position, they can shuffle them around.

5.3.3 Implementation Details

In implementing the game we had to ensure that the chosen platform could cope with the following two important requirements. First, since the game is evaluating per pixel blending algorithms of colors, it should have a framework for handling custom blending functions. Second, the platform of choice should be as generic as possible so that a vast community of players can be reached. In accordance with these two somewhat conflicting requirements, we chose to implement the game as a web application with HTML5/JavaScript handling the game logic, security, design, and data acquisition. The ability of having a custom blending function was achieved through multi-pass rendering utilizing WebGL [5] and GLSL [3] fragment shaders. To handle data storage of collected data, we use a MySQL database server. Whenever there is a valid action from the user, an entry is made into the database through the ASP.NET AJAX framework [1]. Adequate measures were taken to make sure the game server could handle large number of concurrent entries from simultaneous gameplay.

To keep track of high scores and ensure competition among players, we had to decide on a player registration mechanism. We solved this issue with facebook integration for authentication [2]. Due to the popularity of facebook, a large number of players from around the world can simply skip the registration phase and get to the game quickly. Also, a socially powered discussion ensures a more engaging experience. People who have privacy concerns have the option of playing anonymously using a guest account. The welcome page for the game introduces everyone to the concept of gaming with a purpose, presents the story and a tutorial for the game play. Apart from the technical requirements, it also ensures that the person playing the game goes through an Ishihara color blindness test. The game can be accessed by searching for the application "Disguise" in facebook or directly using this URL:

Figure 4. A screenshot of the game.

http://vail.cewit.stonybrook.edu/Projects/HPU/Disguise/. Figure 4 shows a screen-shot from the game.

Finally, to argue the concerns about conducting such color studies over the web, we note that web-based color perception studies have been found to provide comparable results to tests conducted in a lab-based environment. This was determined by Sprow et al. [43] in the context of image quality measurements.

5.4 Correctness: How it computes the target function

We ensure the correctness of the collected data by guiding the behavior of the player towards our goal. We show this by explaining all possible actions a player can take and the feedback the game provides.

Action: Hit an intruder when it is on top of a collector and within the collector's boundary.
Response: Positive reinforcement by increasing the score. Also, the presence of a multiplier encourages consecutive correct answers, making a willing gamer more careful in making choices.

Action: Hit an intruder when it is below the collector and within the collector's boundary.
Response: Negative reinforcement by increasing damage on the collector. No change in the score.

Action: Hit an intruder when it is outside a collector's boundary.
Response: Neutral response. No change in score, no change in collector's health. But, two new intruders are spawned rather than just one to make the space more crowded. This can be a blessing or a curse depending on the game situation. Availability of this action ensures advanced gamers more control over their game, eventually bringing data at a faster rate.

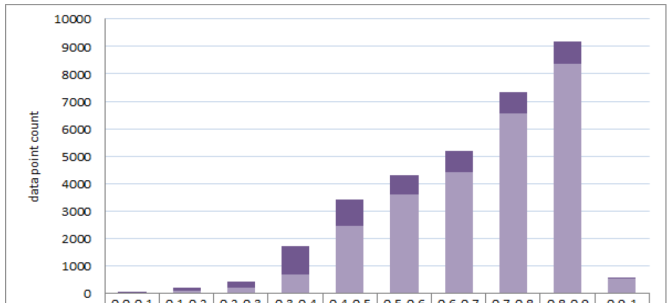
Action: No action.
Response: The game continues. At the end of the lifetime of the intruders they either dissolve (from the bottom layer) or explode (from the top layer). Explosion on top of a collector both damages the collector and causes a negative score. So, the game highly discourages a player who does not do anything.

Thus, the player is always encouraged to judge the ordering of the layers correctly. Most importantly, each click from the player (use of the weapon) solely indicates that according to his/her perception, the intruder color seems to be above the collector color. Thus, every time the player tries to disable an intruder, he is unknowingly acting as a human computation unit, taking as input a pair of semi-transparent colors in predefined order, a blending algorithm and computing whether the composite color preserves perceptual transparency. Thus, we claim that our human computation algorithm computes **evaluate_blend_simple** successfully.

5.5 Evaluation

We evaluate performance and success of the game Disguise from two different perspectives – as a human computation algorithm and as a tool of evaluating visualization algorithms. We claim that as a game, it is fun to play and effective in producing useful data. We also claim that using a purpose-driven game for algorithm evaluation produces better results than other competing methods in terms of cost, data collection speed, quality of results and versatility. To support our claims, we provide an analysis in the following. We compare our results to user studies done in controlled environments and also with recently popular crowd-sourcing techniques.

Playablity: We made our game public on March, 7 2012. So, at the time of this writing, the game can be considered to be very young in age. Yet, within 15 days of its opening we had 261 players playing the game (including guests) generating close to 30,000 data points. Of the 261 players, only 26.7% logged in through facebook, justifying the need to having anonymous access. On average a player played the game for 298 seconds producing 73 data points. Among the registered users, 67.8% returned again to play the game, clearly indicating its attraction. Besides providing all these data, we were also endowed with a lot of encouraging comments and constructive suggestions from the players, signifying their interest in participation and of course their satisfaction. Especially the players, who had high scores, were particularly ecstatic when they went past some very strong high scores. The usage statistics we have here are very modest to say the least. But with the horizon of modern day recreational gaming expanding massively through social platforms, smartphones and tablets, reaching a bigger number of players is very realistic and is all about publicizing it properly. So, a purpose driven game like this is well expected to capture a significant player base providing enough data to serve our purpose.



	0.0-0.1	0.1-0.2	0.2-0.3	0.3-0.4	0.4-0.5	0.5-0.6	0.6-0.7	0.7-0.8	0.8-0.9	0.9-1
Incorrect	2	107	223	1033	976	724	759	791	816	1
Correct	11	92	205	674	2454	3592	4421	6551	8376	539

Figure 5. Stacked column graph showing the distribution of collected data points over different alpha values of the front color.

Data Collection Speed: Our experimental run shows that during gameplay, on average a player produces 14.6 data points per minute. To give an idea how significant this is, just 1,000 players playing the game for 24 hours will already produce the massive count of 21 million data points! To make a comparison, we pick an example of a user evaluation also done to investigate color blending issues [32]. In their study, they evaluated their method via a controlled user study with 72 subjects, producing just 1,728 data points. Considering the history of user evaluations, this number of test subjects was considered very good because of the difficulty of actually finding people in such a large number. But, to actually do a thorough evaluation of the algorithms, we might want to be fancy and cover the whole color and transparency space, giving a requirement of much more than one million data points! This in turn asks the researchers to gather an unrealistic number of test subjects – at a magnitude of hundred thousand. On the other hand, collecting data in millions should be a trivial task for a game due to the large scale collaboration of the internet community. Likewise, the recent progress in crowd-sourcing also allows us to reach that many people for an evaluation of millions of color pairs. But data collection rate in that case will depend a great deal on worker availability, handling of the batches, remuneration offered and a few more factors [22].

Data Collection Cost: Once the game design is done, collecting data is free. The players are using their human brain cycles to compute for us, as a return they are being paid through entertainment. On their part, they perceive that they are playing a game for free; on our part, we are doing user studies without paying anything. Again to compare with other methods, in a traditional controlled user study with $20 remuneration per subject, a test like [32] producing million data points would require close to one million dollars. If we prefer crowd-sourcing, again even with the minimal possible payment of $0.01 per evaluation, a million data points will require $10,000.

Data Quality: A strong argument given in favor of controlled per-person user studies is quality and reliability of the data collected. Hence, historically this has been the favored method for this particular reason. Introduction of crowd-sourcing has definitely provided us with a very cost effective solution of doing the same, but we always have to accept the fact that the result is going to be noisy [22, 24, 38]. The same argument should hold for data collected through human computation because of its internet based data collection scheme. Surprisingly, we found that a game like Disguise provides a very high quality source of data. To understand this finding, it is helpful to explore how human subjects behave in each of these two data collection mechanisms.

Consider a single evaluation of the target function in question. In case of Mechanical Turk (or controlled study), the person will be presented the task of finding the layer ordering in overlapping circles like Figure 2. Irrespective of what his/her answer is, the person is going to be paid at the end of the session. Since, in such setup, the aim of the workers/subjects is to optimize their earnings; they generally do not have the proper motivation to go through the task properly. Because of this, a pair of colors that is a little confusing and requires careful observation has a very high probability of producing random selection on the part of the test subject, especially from Mechanical Turk. On the other hand, in the case of Disguise, the players are giving inputs only because they actually want to play the game, not because they are paid. Their sole objective is to optimize their place on the high score chart and they can do that only by complying with our requirement, hence producing very high quality data points.

Another added advantage is the extent of the collected data. We illustrate this point with a chart showing the distribution of the transparency of colors in the collected data points (Figure 5). The chart shows the count of correct and incorrect predictions by all the players for different alpha value ranges of the front layer color. The number of data points collected with higher alpha value (low transparency) is high because new players take time to cope with the challenge of the game and hence spend more time in playing lower levels. But as they progress through the game, they are provided more and more challenging transparencies. This gives us a unique opportunity of having highly skilled individuals (they are among the top-scorers) judging order perception with lower alpha values. As we can see from the chart, there are players who went into levels where they encountered transparencies in the range 0.0~0.3. Yet they were able to judge the ordering with considerable amount of success. This occurs because they are trying their best, using their eyes and brains to the fullest to survive every single attack from the intruder to beat the highest score. This situation is analogous to a researcher or analyst trying their best to understand a structure or data from a scientific visualization. This level of data collection is not easily possible through traditional user study techniques. A very important point to note here is also that, this argument does not always hold. If the visualization has to be optimized for scenarios where significant cognitive resources have to be allocated to tasks different from very specific low-level tasks, behavior of the gamers generally is not a true evaluator.

Impact of Game design parameters: The game we described in this paper is one of many different possible designs we could have had for exactly the same problem. The type of the game, timing of different events, type and magnitude of positive, negative reinforcement for user actions, type of interaction etc. strongly affect the effectiveness and efficiency of the solution. To illustrate this better, let us consider the impact of a very particular parameter we had to decide on for 'Disguise' – the trajectory of the intruders in the game. Of the many choices we could have made, we decided to use Bézier curves to simulate smooth motions which makes the game more natural and intriguing to play. The movement speed of the intruders is controlled by how long they remain on the playing area before blowing up. Let us call this parameter *intruder presence time*. With a smaller value, we an expect a faster data collection speed given that people continue playing at this rate. On the other hand, a faster speed means shorter time to decide on the layer ordering, making the game hard for new comers. Our initial design of the game had this time time set this time 5 to seconds, irrespective of the level of the game. After publishing the game, a handful of players complained that the speed seemed unreasonably fast, particularly if they were playing the game using a track-pad rather than a mouse or touch screen. Also, because of the faster pace, at the very first level, people who were not still trained enough to play well, clicked in the wrong places producing noisy data. To increase effectiveness of data input, we could have increased the time to some larger value. But this in turn would have reduced the data collection speed and had made the game less challenging for people who are good at it. As a solution, we made the speed adaptive to the player's skill, making the intruders move at a faster pace as they progress through different levels. This had a two-fold impact, first, the new players had a way of learning the game and so, provided better input but at a slower pace. Second, as the player got more and more skilled, they provided input of comparable quality but at a faster rate.

Limitations: Our first effort towards creating purpose driven games for visualization problems exposed some possible limitations of this approach. First, the demography of online gamers is expected to be highly skewed. Unless the game design is done really carefully to avoid influence of player's background and preferences on the game results, this method can only be thought of as an approximation. Second, until there is a formal analytical approach of converting a problem instance into a gaming algorithm, the design of a game will be open ended. Correctness and effectiveness of the gaming algorithm relies very much on the intuition of the designer than on the process itself. Third, designing a game and making it popular takes a considerable amount of time and effort. Until we can find a general technique of mapping any problem instance into a parameter of an already existing game with a large user base, this technique will only be worth pursuing for a limited set of large scale problems.

The discussion so far has made the point that, with our method, it is possible to collect an abundance of reliable data points cheaply. To show how this data can be useful in evaluating the algorithms, we constructed a series of simple plots as shown in Figure 6. Each of these plots shows the number of data points collected and the correctness for different foreground and background combinations. The size of each disk is proportional to the data point count, and its color represents the correctness for that particular combination (blue is predominantly correct, red incorrect). The color scale is linear for the first column of plots (the alpha plots), but non-linear for the others to bring out small differences in the failure cases.

At first glance, we notice that some combinations have less point counts than others. This is simply because these combinations are more complicated and advanced and hence were met (and clicked) by fewer players – the expert high scorers. This statistic is in fact informative on its own as it self-indicates that these combinations are difficult. Next, apart from this more general observation, we can make the following more specific observations:

Observation 1: Irrespective of the algorithm, there is a strong correlation between the alpha value of the layers and order perception. The plots show that an increase in alpha value in the foreground color increases order perception. On the other hand, increasing alpha of background color decreases perception. This observation directly complies with observations from previous works [32, 34].

Observation 2: The algorithms that depend only on color for order perception perform poorly when the hue from foreground and background becomes almost identical. We can observe this from the diagonal in the hue scatterplots. But in the case of blurred background, the player has the additional cue of deformed outlines. This eliminates the trouble with the matching hue.

Observation 3: Hue preserving alpha blending tends to show a higher percentage of correct guesses than basic Porter and Duff blending. But, from the lightness plot of hue preserving blending, it is evident that colors of similar lightness values produce confusion for the players. This behaviour is caused by the fact that the hue preserving technique produces greyish shades in cases of equal lightness.

Observation 4: The overall performance of hue preserving color blending (correct guess: 81.6%) and local alpha blending (84.6%) are comparable. But, the addition of background blurring to local alpha blending introduces significantly more (92.9%) correct guesses from the players.

The analysis presented here is a very partial view of the overall landscape. Dependency between different parameters is much more complex and cannot be explained through two dimensional scatterplots of independent parameters. Doing an analysis that can truly judge the success of a blending algorithm requires a system that can utilize all the data we can collect through this method. This can be an evaluation metric designed for computers or an interactive analysis tool visualizing the high dimensional space to help analysis by a domain expert. Either of them are new research directions by themselves and their details are beyond the scope of this paper.

6 EXTENSIONS

The human computer is a vastly expressive computational platform. The possible ways it can be utilized is only limited by human creativity. We showed how a game like "Disguise" can help evaluate known visualization algorithms. Here, with some examples, we briefly discuss how such ideas can be brought into a more mainstream computation in the visualization pipeline. We hope this will give a better picture of the perspective of this platform and work as a source of inspiration for future researchers in this track.

6.1 Algorithm design assisted by a learning agent

A purpose driven game like "Disguise" can evaluate a given visualization algorithm. Now consider integrating such evaluation mechanism with a learning agent [39]. This gives us an opportunity to build an autonomous system that optimizes the algorithm by itself. It produces a new version of the algorithm that is, by its method of creation, tuned properly for human perception. Figure 7 gives the block diagram of such a learning agent that is made to optimize a blending algorithm for transparency perception. Each block in the diagram shows both the generic elements of a learning system and its counterpart in learning the algorithm. A very challenging part in building such a learning agent is to have a good problem generator which can produce problem instances (in this case a hypothetical color blender) that are both vast enough to allow the system to converge to a globally optimal solution and at the same time restrictive enough to keep the number of computation cycles minimal. Another approach could be the unification of the data acquisition methods in this paper with the data driven approach of Kühne et al. [27] which solved a similar visualization problem.

6.2 Optimizing parameters during the mapping phase

We explained earlier in Section 4.1 that mapping from abstract data to a geometric interpretation can be formulated as an optimization problem. Optimization of such properties (e.g. transfer function) using some local search algorithm (e.g. simulated annealing, genetic algorithm) requires a scoring function that evaluates the strength of the current instance (or set of instances). Since these instances are almost always image data, rather than using approximate evaluators, one can engage human computers by automatically crowd-sourcing evaluation tasks [28, 31]. Theoretically, such hybrid solutions should provide better performance, but at the cost of lost interactivity and added expense. The challenge in such a track remains in finding ways of reusing existing data so that the system eventually becomes sufficiently knowledgeable to drive the visualization without further human intervention.

6.3 Generalization

The particular game we designed solved a very simplified visualization problem, hence was easier to map into an indirect gaming task. Finding such mapping can provide better results but might not be obvious or in some cases impossible for more complicated visualization tasks. Rather than finding a case by case solution for each problem, we could generalize the technique by decoupling the gaming task and the visualization. The game itself could be any popular engaging arcade/puzzle/sports game. The visualization technique under consideration will be then used as an indirect tool for helping the player get better score in the game. Thus the player's performance the game gets mapped to the performance of the particular visualization. This generalization technique is a very early concept and only further investigations can tell us if it can be effective.

6.4 Extreme programming

Another possible extension is extreme programming. We may inject new algorithms (here blending strategies) into the game routinely and instantly monitor if the players improve. We would then age the scores to keep the score board robust to these developments. A good mechanism for this will be the concept of "season" -- every so often at regular intervals the score board would be reinitialized and the previous leaders would become champions of that season and so on. Good scorers would then become multi-season champions.

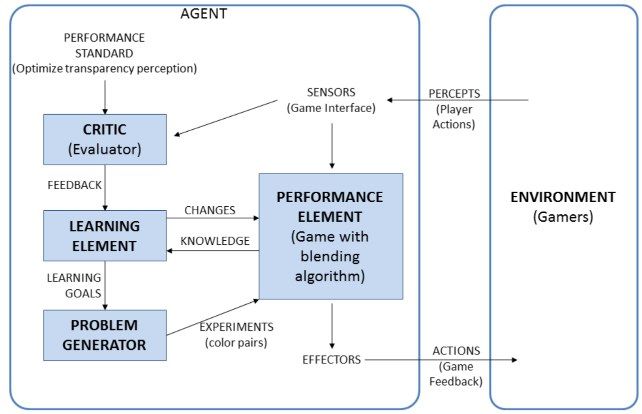


Figure 7. A learning agent to learn a new blending algorithm with the help of a game based evaluation scheme. The diagram shows the common learning agent elements in block letters with their concrete realization within parentheses.

7 CONCLUSIONS

We introduced the concept of computing with humans in the field of visualization. As a demonstration, we presented a detailed design, implementation and evaluation of a purpose driven game that can help evaluate color blending algorithms. According to our experimental data, we found this method to be highly promising and effective. We also introduced some possible ways of extending such a method into the actual computation of rendering of a visualization or designing a new one.

As a future endeavour, we would like to continue discovering new methods of harnessing this new source of computational power to best serve the interest of creating more effective and expressive visualizations. Creating such human computed algorithms is a beautifully creative process and its power is only limited to human imagination. An algorithm like "Disguise" is just an opening to a vast opportunity of research. We are eagerly waiting to see many more solutions like this in future.

ACKNOWLEDGMENTS

The material is based upon work supported by the Department of Energy under Award Number DEOE0000220. This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacture or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favouring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

REFERENCES

[1] (March 2012). ASP.NET AJAX. Available: http://www.asp.net/ajax
[2] (March 2012). Facebook social plugin. Available: http://developers.facebook.com/docs/plugins/
[3] (March 2012). GLSL. Available: http://www.opengl.org/documentation/glsl/
[4] (2012, March 2012). Visualization Pipeline. Available: http://www.infovis-wiki.net/index.php/Visualization_Pipeline

[5] (March 2012). WebGL. Available: http://www.khronos.org/webgl/

[6] L. von Ahn, B. Maurer, C. McMillen, D. Abraham, and M. Blum, "recaptcha: Human-based character recognition via web security measures", Science, vol. 321, pp. 1465-1468, 2008.

[7] Luis von Ahn and Laura Dabbish, "Labeling images with a computer game", presented at the Proceedings of the SIGCHI, Vienna, Austria, pp. 319-326, 2004.

[8] Luis von Ahn, Shiry Ginosar, Mihir Kedia, Ruoran Liu, and Manuel Blum, "Improving accessibility of the web with a computer game", presented at the Proceedings of the SIGCHI, Montreal, Quebec, Canada, pp. 79-82, 2006.

[9] Luis von Ahn, Mihir Kedia, and Manuel Blum, "Verbosity: a game for collecting common-sense facts", presented at the Proceedings of the SIGCHI, Montreal, Quebec, Canada, pp. 75-78, 2006.

[10] Luis von Ahn, Ruoran Liu, and Manuel Blum, "Peekaboom: a game for locating objects in images", presented at the Proceedings of the SIGCHI, Montreal, Quebec, Canada, pp. 55-64, 2006.

[11] Michael S. Bernstein, Greg Little, Robert C. Miller, Bjorn Hartmann, Mark S. Ackerman, David R. Karger, David Crowell, and Katrina Panovich, "Soylent: a word processor with a crowd inside", presented at the Proceedings of UIST, New York, New York, USA, pp. 313-322, 2010.

[12] Jeffrey P. Bigham, Chandrika Jayant, Hanjie Ji, Greg Little, Andrew Miller, Robert C. Miller, Robin Miller, Aubrey Tatarowicz, Brandyn White, Samual White, and Tom Yeh, "VizWiz: nearly real-time answers to visual questions", presented at the Proceedings of UIST, New York, New York, USA, pp. 333-342, 2010.

[13] Chaomei Chen, "Top 10 Unsolved Information Visualization Problems", IEEE Comput. Graph. Appl., vol. 25, pp. 12-16, 2005.

[14] Johnson Chuang, Daniel Weiskopf, and Torsten Moller, "Hue-Preserving Color Blending", IEEE Transactions on Visualization and Computer Graphics, vol. 15, pp. 1275-1282, 2009.

[15] Forrester Cole, Kevin Sanik, Doug DeCarlo, Adam Finkelstein, Thomas Funkhouser, Szymon Rusinkiewicz, and Manish Singh, "How Well Do Line Drawings Depict Shape", ACM Transactions on Graphics (Proc. SIGGRAPH), vol. 28, 2009.

[16] Seth Cooper, Firas Khatib, Adrien Treuille, Janos Barbero, Jeehyung Lee, Michael Beenen, Andrew Leaver-Fay, David Baker, Zoran Popovic, and Foldit players, "Predicting protein structures with a multiplayer online game", Nature, vol. 466, pp. 756-760, 2010.

[17] C. Correa and Ma Kwan-Liu, "The Occlusion Spectrum for Volume Classification and Visualization", Visualization and Computer Graphics, IEEE Transactions on, vol. 15, pp. 1465-1472, 2009.

[18] J. Davis, J. Arderiu, H. Lin, Z. Nevins, S. Schuon, O. Gallo, and Yang Ming-Hsuan, "The HPU", in Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on, 2010, pp. 9-16.

[19] M. Franklin, D. Kossmann, T. Kraska, S. Ramesh, and R. Xin, "CrowdDB: Answering queries with crowdsourcing", presented at the Proceedings of the 2011 ACM SIGMOD, pp. 61-72, 2011.

[20] David Alan Grier, When Computers were Human: Princeton University Press, 2005.

[21] R. B. Haber and D. A. McNabb, "Visualization idioms: A conceptual model for scientific visualization systems", Visualization in Scientific Computing, pp. 74-93, 1990.

[22] Jeffrey Heer and Michael Bostock, "Crowdsourcing graphical perception: using mechanical turk to assess visualization design", presented at the Proceedings of the 28th international conference on Human factors in computing systems, Atlanta, Georgia, USA, pp. 203-212, 2010.

[23] Chang Hu, Benjamin B. Bederson, Philip Resnik, and Yakov Kronrod, "MonoTrans2: a new human computation system to support monolingual translation", presented at the Proceedings of the 2011 annual conference on Human factors in computing systems, Vancouver, BC, Canada, pp. 1133-1136, 2011.

[24] Panagiotis G. Ipeirotis, Foster Provost, and Jing Wang, "Quality management on Amazon Mechanical Turk", presented at the Proceedings of the ACM SIGKDD Workshop on Human Computation, Washington DC, pp. 64-67, 2010.

[25] Aniket Kittur, Ed H. Chi, and Bongwon Suh, "Crowdsourcing user studies with Mechanical Turk", presented at the Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems, Florence, Italy, pp. 453-456, 2008.

[26] Robert Kosara and Caroline Ziemkiewicz, "Do Mechanical Turks dream of square pie charts?", presented at the Proceedings of the 3rd BELIV'10 Workshop: BEyond time and errors: novel evaLuation methods for Information Visualization, Atlanta, Georgia, pp. 63-70, 2010.

[27] Lars Kühne, Joachim Giesen, Zhiyuan Zhang, Sungsoo Ha, and Klaus Mueller, "A Data-Driven Approach to Hue-Preserving Color-Blending", IEEE Transaction on Visualization and Computer Graphics, 2012.

[28] Anand P. Kulkarni, Matthew Can, and Bjoern Hartmann, "Turkomatic: automatic recursive task and workflow design for mechanical turk", presented at the Proceedings of the 2011 annual conference extended abstracts on Human factors in computing systems, Vancouver, BC, Canada, pp. 2053-2058, 2011.

[29] Edith Law and Luis von Ahn, Human Computation: Morgan & Claypool Publishers, 2011.

[30] Edith Law, Luis Von Ahn, Roger B. Dannenberg, and Mike Crawford, "TagATune: A Game for Music and Sound Annotation", ISMIR, pp. 361-364, 2007.

[31] Greg Little, Lydia B. Chilton, Max Goldman, and Robert C. Miller, "TurKit: tools for iterative tasks on mechanical Turk", presented at the Proceedings of the ACM SIGKDD Workshop on Human Computation, Paris, France, pp. 29-30, 2009.

[32] Wang Lujin, J. Giesen, K. T. McDonnell, P. Zolliker, and K. Mueller, "Color Design for Illustrative Visualization", IEEE Transaction on Visualization and Computer Graphics, vol. 14, pp. 1739-1754, 2008.

[33] Adam Marcus, Eugene Wu, David Karger, Samuel Madden, and Robert Miller, "Human-powered sorts and joins", Proc. VLDB Endow., vol. 5, pp. 13-24, 2011.

[34] F. Metteli, "The perception of transparency", Scientific American, vol. 230, pp. 91-98, 1974.

[35] Chan Ming-Yuen, Wu Yingcai, Mak Wai-Ho, Chen Wei, and Qu Huamin, "Perception-Based Transparency Optimization for Direct Volume Rendering", Visualization and Computer Graphics, IEEE Transactions on, vol. 15, pp. 1283-1290, 2009.

[36] Catherine Plaisant, "The challenge of information visualization evaluation", presented at the Proceedings of the working conference on Advanced visual interfaces, Gallipoli, Italy, pp. 109-116, 2004.

[37] Thomas Porter and Tom Duff, "Compositing digital images", SIGGRAPH Comput. Graph., vol. 18, pp. 253-259, 1984.

[38] Joel Ross, Lilly Irani, M. Six Silberman, Andrew Zaldivar, and Bill Tomlinson, "Who are the crowdworkers?: shifting demographics in mechanical turk", presented at the Proceedings of the 28th of the international conference extended abstracts on Human factors in computing systems, Atlanta, Georgia, USA, pp. 2863-2872, 2010.

[39] Stuart Russel and Peter Norvig, "Intelligent Agents", in Artificial Intelligence: A Modern Approach, Third Edition ed: Prentice Hall, 2010, p. 54.

[40] Petr Šereda, Anna Vilanova, and Frans A. Gerritsen, "Automating Transfer Function Design for Volume Rendering Using Hierarchical Clustering of Material Boundaries", Eurographics/IEEE VGTC Symposium on Visualization (EuroVis), pp. 243-250, 2006.

[41] M. Singh and BL. Anderson, "Perceptual assignment of opacity to translucent surfaces: The role of image blur", Perception, vol. 31, pp. 531-552, 2002.

[42] M. Singh and BL. Anderson, "Toward a perceptual theory of transparency", Pschol Rev, vol. 109, pp. 492-519, 2002.

[43] Iris Sprow, Zofia Bara´nczuk, Tobias Stamm, and Peter Zolliker, "Web-based psychometric evaluation of image quality", Image Quality and System Performance VI (Proc. SPIE), vol. 7242, 2009.

[44] D. G. Stork, "The Open Mind Initiative", IEEE Intelligent Systems & Their Applications, vol. 14, pp. 19-20, 1999.

[45] Jarke J. Van Wijk, "The value of visualization", IEEE Visualization, pp. 79-86, 2005.

[46] Zheng Ziyi, N. Ahmed, and K. Mueller, "iView: A Feature Clustering Framework for Suggesting Informative Views in Volume Visualization", Visualization and Computer Graphics, IEEE Transactions on, vol. 17, pp. 1959-1968, 2011.

Figure 6. Visualization to evaluate the performance of different blending algorithms for transparency perception. For each of the algorithms we produce three scatterplots concerning variation in alpha value, hue and lightness of the colors. Each of the plots shows the number of data points collected and the correctness for different foreground and background combinations. The size of each circle is proportional to the data point count. The color of the circle represents the correctness for that particular combination.