Visualizing Real-Time Strategy Games: The Example of StarCraft II

Yen-Ting Kuan, Yu-Shuen Wang, Jung-Hong Chuang
National Chiao Tung University

ABSTRACT

We present a visualization system for users to examine real-time strategy games, which have become very popular globally in recent years. Unlike previous systems that focus on showing statistics and build order, our system can depict the most important part – battles in the games. Specifically, we visualize detailed movements of armies belonging to respective nations on a map and enable users to examine battles from a global view to a local view. In the global view, battles are depicted by curved arrows revealing how the armies enter and escape from the battlefield. By observing the arrows and the height map, users can make sense of offensive and defensive strategies easily. In the local view, units of each type are rendered on the map, and their movements are represented by animation. We also render an attack line between a pair of units if one of them can attack the other to help users analyze the advantages and disadvantages of a particular formation. Accordingly, users can utilize our system to discover statistics, build order, and battles, and learn strategies from games played by professionals.

Keywords: real-time strategy games, StarCraft, game visualization, trajectories

1 INTRODUCTION

Real-time strategy (RTS) games have become increasingly popular in recent years. The games in this category are generally played by two players (teams). The goal is to build a nation and defeat the other nation. Players have to consider many perspectives such as managing resources, buildings, techniques, and military forces to win a game. The strategies are very complex, and players must always make correct decisions immediately throughout the game. Among the RTS games, Starcraft II (SC2 in short) is the most representative work. It has sold several million copies internationally. In addition, the game's world championship is held annually and the broadcasts can be viewed in many countries through the Internet and on television. Furthermore, after defeating human beings in the game Go, a research team at Google is training an artificial intelligence, DeepMind, to play SC2 in an effort to defeat human beings, as well. All of these facts testify to the great fame of this game throughout the world.

Although learning to play SC2 is easy, learning to be a master is difficult. Players have to learn effective strategies by watching game replays played by professionals using the official replay system. All details, such as when and where the buildings are constructed and how the armies attack the enemies, can be observed. In other words, there is no difference between watching a replay and an on-going game. However, like the real world, events in RTS games occur simultaneously, and resources, techniques, units, and battles are all correlated. When watching a replay by using the official system, events appear in the game over time and users may not be able to understand the cause of a defeat immediately. Under this circumstance, they have to control the time slider and watch the game clip over and over again. In addition, the official system shows only a small part of the map in a replay. Although users can pan the view window interactively when using the system, they are very likely to miss important events, and have no way of discerning why the player was defeated.

Besides the official replay system, Scelight [3] and GGTracker [7] are two visualization tools commonly used by SC2 players (Figure 1). Both tools apply line charts to depict certain statistics such as population, used and unused resources over time so that users can understand growth and decline at each perspective of the nations. In addition, Scelight provides users with the build order of each nation; and GGTracker plots units and buildings by scatter points on a small map. The disclosed information is helpful for users to obtain a quick overview of a game. However, although battles play a critical role in RTS games, they are not visualized by the tools. A nation with stronger forces may lose the game if the player adopts an ineffective offensive strategy. Users have to understand the configuration of units, the ground height, and how the battles take place if they attempt to know the reason of a defeat or victory. Since the tools do not visualize battles in SC2 games, users have to seek answers by watching replays using the official system.

We present an interactive system to visualize SC2 games while preventing the above-mentioned drawbacks. The goal is to help users learn strategies from replays. The system consists of 1) color-coded charts depicting statistics of nations over time; 2) a build order showing the pattern of production, technique, and resource management; 3) a small map conveying the distribution of units and buildings; 4) a ThemeRiver [5] representation of deaths over the time that battles occur; and most importantly, 5) a large map that can show users how the battles take place. We design a global view and a local view for the map and help users make sense of a battle in a coarse-to-fine grained manner. In the global view, we apply a battle glyph to convey the loss of forces; a heat map to disclose the distribution of armies; and curved arrows to depict the movements of these armies. The designs can effectively describe a long time span of a battle and prevent visual clutter when visualizing it in a limited area. In the local view, we plot every unit on the map and convey their movements by animation. Considering various attack distances of units in different types, we draw a half-transparent attack line between a pair of units if one of them can attack the other. The color of an attack line indicates the nation. The density of these lines assist users to analyze the advantages and disadvantages of a formation. As a result, by investigating views provided by our system, users can make sense of all details and learn effective strategies from games.

We visualize several classical game replays by the presented system to verify its feasibility. The insights are disclosed and described in the results section. We also conducted a user study with 20 participants who had played SC2 games for 0.5-7 years. During the study, they were asked to answer the predefined questions by using the compared and the presented systems, respectively. We recorded the time they spent to answer questions and sought their feedback. They gave the presented system quite positive comments after using it. They also mentioned that the system was not only convenient to examining a game but also effective at teaching strategies.

2 RELATED WORKS

Game visualizations. Game visualizations have been studied for many years because they are helpful to playing and designing games. The visualizations are purpose-oriented and can be categorized into on-line and off-line systems [20]. The works of Wallner and

Figure 1: Top and bottom are the screenshots of Scelight [3] and GGTracker [7], respectively

Kriglstein [19, 21] allow game designers and managers to define states in a game and depict player behaviors by visualizing the transitions of the states. Medler et al. [11] presented a visualization system for a third-person shooting game called Dead Space 2. The system helps players review strategies and player performances in games. Liu et al. [8] introduced a system to depict how players play a puzzle game by using graphs. The system reveals the advantages and disadvantages of the stage design and helps designers to create new stages. In addition to visualizing player behaviors in a stage, the work presented by Thawonmas et al. [17] attempts to visualize how MMORPG (massively multiplayer online role-playing games) players select stages and social contact with each other. They applied graphs to show relation between the frequently appearing positions and the categories of players, in order to refine the business model.

Data analysis and visualization for RTS games. Almost all RTS game players attempt to discern the keys to winning a game because of its complexity. Low-Kam et al. [9] applied sequential pattern mining to the build orders of a game dataset. The presented method demonstrates the scalability and efficiency in discovering unexpected game strategies. Avontuur et al. [1] analyzed game statistics of players at different levels and attempted to identify the relation between game levels and strategies. In addition to winning a game, Weber and Mateas [22] and Hsieh and Sun [6] presented methods to predict defeat or victory of a player in a game. They applied machine learning techniques to train models based on build orders in a large collection of game logs.

In addition to data analysis, players can review games and learn strategies by using visualization tools. Gagne et al. [4] visualized the logs of a game called Pixel Legions by drawing the trajectories of units. They randomly selected 20% of the trajectories from a game log, and then rendered them with half-transparency to achieve the aim. Considering that SC2 games usually involve much more complex behavior of units in a larger scale than Pixel Legions, visualizing SC2 games by using the method would inevitably result in visual clutter and omit important events. Wender et al. [23] introduced a trace-based system for SC1 and show the game logs by text and line charts. Belicza [3] presented a tool called Scelight to depict statistics of SC2 games by line charts. The tool also presents a time table to indicate the build order of techniques and buildings constructed by players. However, the tool lacks a map view so that users cannot observe how a battle occurs when using it. Similarly, Joerg [7] developed a tool called GGTracker to visualize SC2 games by line charts. This tool provides users with a small map and shows the distributions of units and buildings by scatter points belonging to respective nations. Because of the very small map and non-differentiation of units and even buildings, GGTracker is unable to depict strategies applied on the battlefield.

RTS games AI. Artificial intelligence has been successfully applied to RTS games. Ontañón et al. [12] extracted behavioral knowledge from experts and implemented a case based behavior generator to play RTS games. Synnaeve and Bessière [15] discovered the game replays of SC1 and represented the armies' components by the Gaussian mixture models. This compact representation enables AI to efficiently assess situations and adapt strategies in RTS games. Afterward, they decomposed RTS games AI into strategy, tactics, and micromanagement, and simplified the complex AI problem by considering hierarchical levels and sequentiality of decisions. The Bayesian models are then applied to optimize the probability that can beat the opponent [16]. Considering that establishing military forces demands an effective build order, which is a sequence of collecting resources, constructing buildings, researching techniques, and training units, Ballinger et al. [2] presented a fitness function to measure the performance of a build order and developed a coevolutionary approach to compute an effective one for RTS games. In addition, since the simulator is crucial for training an RTS games AI, Uriarte and Ontañón [18] presented a combat model for SC1 and learned the parameters from replay data. For all details related to RTS games AI, we refer readers to the survey paper [13] since the presented study does not focus on this field.

3 Design Methodology

3.1 Background

SC2 is an RTS game, in which the game progresses in every second rather than in turns. The goal is to build a nation based on the selected race and establish the military to eliminate the opponent. There are three races in the game: protoss, terran, and zerg. Each race has its own techniques, buildings, and units. Overall, units of a protoss nation are few, but strong; units of a zerg nation are many, but weak; and units of a terran nation are in between the two.

Each player has a base and several workers in the beginning of a game. They command workers to collect minerals and vespene gas around the base that will be used to construct buildings, train units, and research techniques. Generally, buildings are constructed to serve several purposes such as to produce units, upgrade techniques, or provide static defences. Units are trained and organized armed forces that can fight on the battlefield. Techniques are researched to enhance the strength of units such as increasing speed, attack, and defence. The consumption of minerals and vespene gas depends on the strategy applied to defeat the opponent. In addition, limited by the speed of resource collection and the prerequisites to research specific technologies or build specific advanced units, players have to make decisions in a sequence of upgrades while managing resources. Typically, the sequential pattern of production, research, and training is called "build order", which is an important key to establish a strong military. However, owing a strong military does not guarantee winning the game because strategies on a battlefield is also important. Players have to consider the formation, the prevailing of units, and the ground height while leading armies across a battlefield against enemy forces. In short, SC2 is a very complex RTS game. Resource management, buildings, techniques, units, and offensive and defensive strategies are all critical in defeating the opponent. We refer readers to Wikipedia [1] for details.

3.2 Datasets

We downloaded game replays of SC2 from Spawning Tool[2]. These replays record the commands of games played by professionals, including when and where the buildings are constructed and destroyed, when and what techniques are researched, and the status of unit throughout the game. In other words, the replay encodes all details

[1] https://en.wikipedia.org/wiki/StarCraft II: Wings_of_Liberty
[2] http://lotv.spawningtool.com/

Figure 2: Left and right are the status and the battle views in the presented visualization system, respectively. When users select a time span in the status view, the system would switch to the battle view for the examination of offensive and defensive strategies. Color shadings and arrows are used to depict the movements of armies. (a) The status view shows the relative economic strengths of two nations over time. (b) The build order indicates the time that the techniques and buildings are constructed. (c) The death ThemeRiver shows the number of deaths in battles over time. (d) The status bar indicates the transfer function and the related information in the game play. (e) The small map conveys the distributions of armies and buildings. (f) The map is used to reveal geographic features and show the process of a battle. (g) The line charts show the strengths of armies in each type. (h) The context view of the map and the current techniques owned by the two nations.

of a game and can be decoded for game reconstruction. We use a tool called SC2Reader[3] to parse the replay files and implement an SC2 simulator to reconstruct the game. Specifically, we compute the moving trajectory of each unit by using the A* search [14]. To ensure a correct reconstruction, we visually compare the reconstructed games in this study and the corresponding game videos on YouTube, and did not observe any differences.

3.3 Requirement Analysis

There are two important aspects in playing SC2 games: 1) establishing a strong military that can prevail against the enemy units; and 2) applying offensive and defensive strategies on the battlefield to defeat the opponent. To help users learn strategies from these two aspects, we discussed with several SC2 players and formulated a set of requirements to guide the visualization design. Specifically, the system has to help users:

R1 ***build the global concept of a game.*** SC2 is a type of RTS games that has very complex strategies. Decisions related to buildings, techniques, units, and resource collections are all correlated and would greatly effect the ending of a game. While watching a replay using the official system, users have to memorize the commands given by players and build the concept of the game in mind by themselves. Therefore, a visualization interface that can depict global trends related resources, buildings, techniques, and units are helpful. Users can compare what had been done by the players and observe the influence of the decisions in the subsequent developments in a game.

R2 ***quickly identify important events in a game.*** Important events do not frequently occur in a game. When watching a replay using the official system, users have no idea the time that a crucial event occurs and have to continue retrieving the event by dragging the time slider. They also could miss the events because only a small part of the map can be displayed on the screen. Therefore, an interface that can help them quickly identify important events is necessary.

R3 ***learn how to establish a strong military within a short period of time.*** When users observe the early stage of a game, they are interested in knowing the build order executed by an expert under the resource limitation. Discovering the build order is important because prerequisites should be satisfied before upgrading specific techniques, constructing specific buildings, and training advanced units.

R4 ***learn how to choose a good battlefield.*** Units standing at high ground have advantages against units standing at low ground. In addition, some areas on the map are restricted and only air forces can cross. Therefore, forcing the enemies to fight at low ground and preventing them from escape are important. Users would be interested in discovering how experts utilize geographic features to win a battle.

R5 ***discover the formation and the movements of units.*** Users would like to discover how experts control units during a battle because they are the fundamental elements of tactics. For example, suppose that a group of units A are surrounding the enemy units B, then units A would enjoy the advantages in attacking units B. Users are interested in learning how to perform such a tactics.

3.4 Visualization Interfaces

We provide users with a visualization system that can satisfy the above-mentioned requirements. Because most SC2 games are played by two players, before the introduction of our designed interface, we point out that we use red and blue to represent the data of each nation (player). Other information such as races, players' names, and defeat or victory can be obtained in the bottom status bar.

The developed visualization system is composed of a status view and a battle view for users to examine iteratively and alternatively. The status view shows the time varying statistics of each nation. Specifically, it 1) conveys populations, used and unused resources, and military strengths at each time span; 2) indicates when and what techniques are developed; and 3) reveals the number of dead over time. We also render a small map on the left of this status view and plot scatter points to approximately indicate the distributions of units. On the other hand, the battle view is designed to help users discover offensive and defensive strategies on the battlefield. Particularly, it shows how armies enter and escape from a battlefield, and reveals how geographic features affect a battle. Since users may zoom in to the map for a close examination, in this battle view, we also render a

[3]https://github.com/GraylinKim/sc2reader

context view, the amounts of units, and the techniques that can be used in the battle for users to consider.

3.4.1 Status View

Figure 2 (a-c) shows the three views depicting the time varying statistics of respective nations (**R1**, **R2**). They indicate resources, build order, and the numbers of dead over time, respectively. The three views share the same time coordinate system that can be observed at the bottom. In addition, our system renders a vertical line across the three views when the cursor is moved over to help users align the information. The view of resources is composed of color-coded charts, in which the color is determined according to the transfer function (Figure 2 (d)). Namely, blue and red indicate that one of the nations is stronger than the other over 30% on the indicating attribute, and white means that the nations have equal strengths. The continuous shades are computed by linear interpolation. The view of build order (Figure 2 (b)) contains various icons representing the developed techniques and the constructed buildings (**R3**). The horizontal coordinate of an icon indicates the time that the represented structure or technique was accomplished. The vertical coordinate, however, is allocated only to prevent occlusions. Theoretically, the vertical coordinate can be unlimited. But we stack the icons to at most four rows to achieve a compact visualization. Users are allowed to expand the vertical coordinate if necessary. If the collision still occurs, we apply the spring algorithm to slightly modify icons' horizontal coordinates and prevent the occlusion. However, we experimentally observe that allocating four rows in this view is enough to depict the build order because limited resources and manual operations would prevent too many techniques and buildings to be accomplished at the same time. In addition to the coordinates, the icon representing a building is labelled by *x(y)* at its bottom left, where *x* and *y* indicate the newly built and the total numbers of buildings, respectively. The icon representing a technique has no labels because a technique can only be developed once in a game. The number of dead is visualized by the ThemeRiver [5] (Figure 2 (c)). The color depicts the nation, and the heights indicate the numbers of dead.

Our system shows a small map at the bottom left of this status view (Figure 2 (e)) . When the cursor is moved over the time varying data, the positions of units at the specified time span are indicated by scatter points on this map. We also list the exact number of units of each type to help users evaluate the military strengths.

3.4.2 Battle View

Figure 2 (f) shows a map to describe the process of a battle. When users observe an interesting event in the status view such as a large number of dead, they can select a time span in the ThemeRiver and our system will switch to the battle view (**R2**). Because units standing on high ground have advantages when fighting those on low ground, we render a height map rather than the original game map in this view (**R4**). The regions close to green are higher than the regions close to white; and the regions in gray are restricted and only air forces can across. In addition, we plot the positions of minerals and vespene gas, so that users can know where the resources are mined by the nations.

Users can pan and zoom the map to closely examine the details of a battle. To help users keep the global concept of a battle and understand the relative positions of units, at the left of this battle view (Figure 2 (h)) we draw a context view and indicate the closely examined area by a bounding box. In addition, we apply a line chart to depict the military strength of units of each type over time, and some icons to indicate respective techniques that can be used during battle (Figure 2 (g)). It is worth noting that the height of a line chart does not indicate the number of units, but rather the military strength. The design is made because units of different types are of different strengths, which can be computed according to the resources demanded for training [3, 7]. In addition, the height of each line chart indicates the strength normalized by the maximum strength in the battle. We also list the exact number of units at the right of each line chart when the cursor is moved over to specify a time span because the number is important to professional players.

3.5 Hierarchical Battle Visualization

We visualize the battle in a hierarchical manner. When users zoom the map out, where drawing individual units must result in serious visual clutter, we visualize the battle by shadings, glyphs, and trajectories. When users zoom the map in, where the space is large, we visualize the movement of every unit by animation (**R5**).

Considering that users only select a time span before the visualization, the system has to identify the units involved with a battle. To achieve the aim, we consider the units that die in the selected time span and determine the battles according to distances. Specifically, each pair of dead units is connected by an edge if the moving distance between their places of death is smaller than a threshold α. Since not all units that participate in a battle will die, we also consider the survivors involved with the battle as well if they are spatially and temporally close to the deaths. In other words, for each unit that dies in a battle, we set a β-second time window immediately before the death. All of the survivors, where their spatial and temporal distances to a death are smaller than α and β, respectively, are involved and connected to the death by edges. Afterwards, each disjoint set on the map is treated as a battle. In our implementation, we set $\alpha = 15$ because this value is the maximum distance for one unit to attack the other; and $\beta = 4$ because the value is the maximum time for one unit to move α distance on the map.

3.5.1 Global View of a Battle

To visualize a battle at a global view, we shade the map in red and blue to represent the positions of units belonging to respective nations. We also draw trajectories to convey their consecutive movements, as well as battle glyphs to indicate where the battles take place and the growth and decline of military strengths over time, as illustrated at right. This battle glyph can be considered as a narrow (square) version of a ThemeRiver, where the heights of the left and right boundaries indicate the military strengths before and after the battles, respectively. Similar to other designs in our visualization system, blue and red of a battle glyph represent different nations. However, we further split the colors into dark and light versions to indicate ground and air forces, respectively. Given that the region around a battle often is shaded in blue or red to represent the armies, we add a white border to the glyph to prevent it from becoming invisible on the map.

Battle glyph.

To abstract a battle, we sample the position of every unit involved with the battle per 0.1 second and form a point set. Each point contains the position itself and whether a unit dies at that position. Then, we merge the points iteratively by agglomerative hierarchical clustering and create a tree. Each leaf node on the tree corresponds to a point on the map, and the other nodes are the merged results. To achieve a clean and compact visualization, we partition the tree to upper and lower parts and visualize only the upper nodes on the map. Specifically, our system traces the tree downward from the root and back traces whenever it encounters a node that: 1) does not contain any deaths or 2) is spatially too close to its parent node (i.e., the distance on the map is smaller than γ). For the traced node (i.e., on the upper part of the tree), in which casualties appear, we draw a battle glyph at the corresponding position on the map to represent a battlefield. Since the level of detail is proportional to the zoom level of a map, we let users control the threshold γ. In addition, because the hierarchical clustering may take a long time when merging points of a large set, we apply the k-means algorithm

Figure 3: The positional constraints prevent the curved arrows from occluding with each other and passing through obstacles. (Left) The original curved arrows. (Right) The revised version.

(k=1000) to the set prior to the hierarchical clustering to reduce the computation cost.

Each node on the tree corresponds to a location on the map. The goal is to visualize units' movements by drawing nodes and arrows. To achieve the goal, for each pair of nodes n_a and n_b, if there are units moving from node n_a to n_b, we draw a curved arrow between the two nodes. The shape of this curved arrow is computed by approximating a Bèzier curve to the units' moving trajectories so as to reduce information distortion. This property is very important to visualizing movements of ground forces because they often have to make a detour to avoid obstacles. In addition, the width of a curved arrow is determined based on the total strength of units that move in between a pair of nodes. Users also can enlarge the widths of all arrows by tuning a common parameter ρ if necessary.

To compute a Bèzier curve to represent the units' movements, we consider each sampled point $\mathbf{p}_i^t$ on every moving trajectory i and parameterize the point to $t \in [0,1]$ according to the relative position from node n_a to node n_b. Then, we solve an objective function

$$\Omega_b = \sum_{i \in T} \sum_t \omega_i^t \left| \sum_{\ell=0}^{d} h_i^{t,\ell} \mathbf{q}^\ell - \mathbf{p}_i^t \right|^2, \tag{1}$$

where $h^{t,\ell} = \binom{d}{\ell} (1-t)^{d-\ell} t^\ell$ is an interpolation coefficient, $\mathbf{q}^\ell$ is the control point of a Bèzier curve, d is a degree of freedom (we set $d=3$ in our implementation), and ω_i^t is a weight indicating the proximity of point $\mathbf{p}_i^t$. Considering that a Bèzier curve is only an approximation of the moving trajectory, the computed arrow may deviate from the positions of nodes n_a and n_b due to the smoothness constraint. To prevent misunderstanding, we set a large weight to ω^t if $t > 0.9$ or $t < 0.1$ so that the sampled points close to nodes n_a and n_b can be better approximated. In addition, a curved arrow representing the moving trajectories of ground forces may pass through an obstacle due to the smoothness constraint. Let $\hat{\mathbf{p}}_i^t = \sum_{\ell=0}^{d} h_i^{t,\ell} \mathbf{q}^\ell$ be the approximated position of $\mathbf{p}_i^t$. We also set ω_i^t to a large value if $\hat{\mathbf{p}}_i^t$ is on an obstacle. Figure 3 shows a comparison with and without the constraints.

Curved arrows are, in general, independent of each other and can be computed in individual steps. However, the obtained arrows could have some occlusions and lead to visual clutter. Therefore, we solve all curved arrows simultaneously and prevent the approximated points $\hat{\mathbf{p}}_i^t$ from being too close to each other. Let $\hat{\mathbf{p}}_i^t$ and $\hat{\mathbf{p}}_j^s$ be the points on curved arrows i and j, respectively. To maintain a minimum distance between them, we add an energy term

$$\left| (\hat{\mathbf{p}}_i^t - \hat{\mathbf{p}}_j^s) - \varepsilon (\mathbf{p}_i^t - \mathbf{p}_j^s) \right|^2 \tag{2}$$

to the objective function. Note that preventing the point $\hat{\mathbf{p}}_i^t$ from locating at an obstacle and from being too close to another point $\hat{\mathbf{p}}_j^s$ is achieved by solving inequality constraints, we minimize the objective function in an iterative manner. In the beginning, we suppose

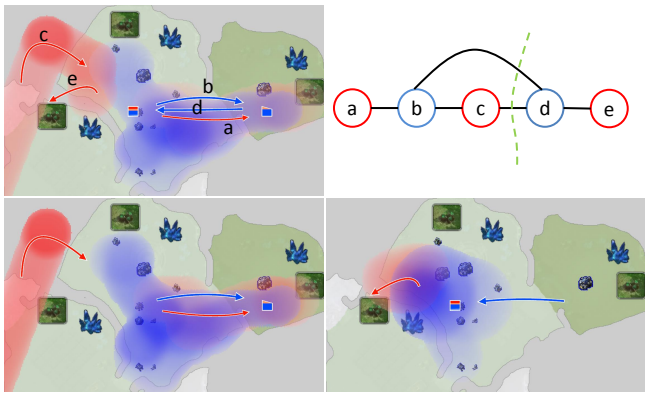

Figure 4: (Top left) The curved arrows may form a cycle and mislead users by the order. (Top right) Our system partitions the set of curved arrows into groups based on the time coordinates to prevent the misleading. (Bottom) The consecutive key frames.

that all inequality constraints are satisfied and computes the curved arrows. Then, it detects whether the obstacle and collision problems occur to the approximated point $\hat{\mathbf{p}}_i^t$, and updates the weight or adds additional energy terms to the objective function. The iteration stops when all inequality constraints are satisfied. We refer readers to [10] for the details of constrained optimization.

Drawing all curved arrows of a battle potentially causes cognitive problems due to temporal ambiguity. A cycle appears when two curved arrows have opposite start and end locations on a map, and users usually cannot discern which arrow is earlier than the other. Figure 4 shows an example of this. Therefore, we draw the arrows in steps to prevent this problem. Specifically, we detect cycles among the curved arrows and sort the arrows according to the time spans of the start points. Then, a graph with each node representing a curved arrow is built and edges are added to connect nodes if the curved arrows are temporally adjacent or appear to have a cycle, as illustrated in Figure 4 (top right). We define the cost of each edge by

$$C_{ij} = \begin{cases} 100 & \text{if a cycle is composed of curved arrows } i \text{ and } j \\ t_d & \text{otherwise} \end{cases}, \tag{3}$$

where t_d is the temporal distance (seconds) of the two arrows. In each step, the goal is to solve the ambiguity problem by partitioning the graph into two sub-graphs in which the summed cost is minimal. We apply the linear programming algorithm to find a cut passing through edges that have large costs. The process stops when cycles disappear or when the number of sub-graphs reaches a threshold. Thus, the curved arrows in consecutive sub-graphs are sequentially visualized to prevent the temporal ambiguity problem. Our system also fades in and out of the arrows when transiting consecutive arrows to enhance the continuity impression of armies' movements.

3.5.2 Local View of a Battle

At a local view, every unit on the map is visualized by a small image and its motion is represented by an animation so that all details can be revealed. Considering the different attack distances of units, and the attack limitations between ground and air forces, we draw an attack line between units if one of them can attack the other. Specifically, denote by S_b and S_r the units belonging to nations represented in blue and red, respectively. If unit S_b is able to attack unit S_r, we draw a half transparent blue line. Regarding the opposite situation, we draw a half transparent red line. The degree of transparency is determined according to the strength of attack. In other words, if a region is mostly overlapped by lines in a color, even though the military strengths of the two nations are close, the result of this

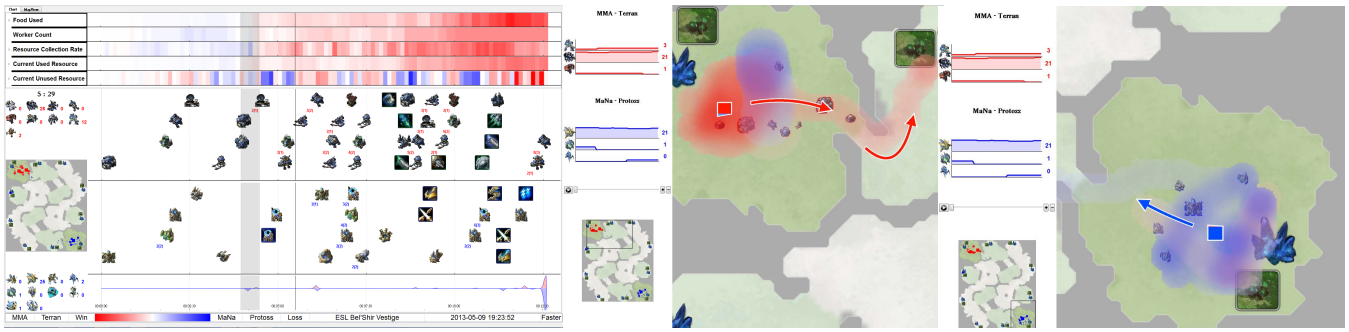

Figure 5: (Left) Users first examine the status view, build order, and the death themereiver to seek interesting events. Then, they select a time span and switch to the battle view to observe details. (Middle and right) In this fast attack event, the terran base was garrisoned by some units, but the protoss base was not. Accordingly, many protoss workers were eliminated and the overall development after the attacks slowed down.

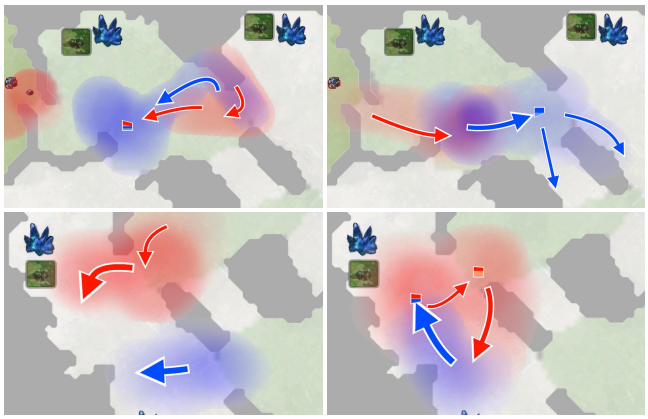

Figure 6: Top and bottom show a successful and a failure examples of surrounding enemies. Key frames are selected and sequentially displayed from left to right in each example.

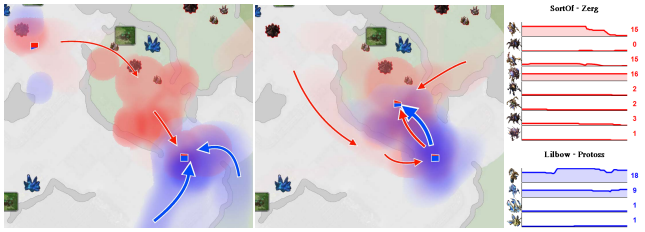

Figure 7: Left and middle show the sequential key frames and right indicates the strengths of units in each type. In this example, the armies of zerg are subdued by the armies of protoss. The protoss won the game although its armies were surrounded and had geographic disadvantages at the early stage of the battle.

battle is clear. We point out that these attack lines are particularly useful to novice players because they are not good at commanding units of different types and utilizing geographic features to gain advantages over the enemy in a battle. The feedback obtained from the conducted user study verifies this argument.

4 Results

We have implemented the presented system using C++ and run the program on a desktop PC with Intel Core i7 3.0 GHz CPU. Users can examine SC2 games interactively by using our system, except when they select a time span and switch the view to observe battles. In our experience, users have to wait approximately 2-5 seconds because the system has to locate positions of battlefields and compute curved arrows by hierarchical clustering and optimization, respectively. The computation time is mainly based on the number of units in a battle.

To study a game replay by using our system, users first discover the statistics, build order, and death ThemeRiver in the status view. If they have observed events of interest, they can select a time span and then examine the distributions of units on the small map or switch to the battle view for further examination. For the example shown in Figure 5, red and blue indicate terran and protoss, respectively. The unused resource of protoss was slightly more than that of terran before the highlighted time span. However, the situation then changed after that because most of the color-coded charts turn red. Meanwhile, the density of icons in the build order view indicates a similar situation, in that the terran had many more buildings and techniques than the protoss, which also implied that the terran armies were stronger than the protoss armies. Since the ThemeRiver below indicates a small number of dead around the turning point, users can select the time span and then switch to the battle view to find out the answer. As can be seen, both the terran and protoss adopted the fast attack strategy. On the one hand, a certain amount of terran armies were garrisoned around the base so that the fast attack from protoss could be stopped quickly. On the other hand, the protoss base was not defended, and several workers were killed during the fast attack. Given that the resource collecting speed were slowed down, the development of the protoss nation was affected.

We present more case studies below to explain how insights and strategies of SC2 games are identified by using our visualization system, particularly, through the battle view.

Case 1: Surround. Figure 6 top shows an example in which armies successfully escaped from the place surrounded by enemies. In the beginning, the blue armies gathered outside of the red base and prepared for the attack. However, they had been detected by the red armies garrisoned at the right. When the blue armies moved left to attack the base, they were surrounded by the red armies. Despite the disadvantages, the blue armies evacuated to the bottom right region separately and immediately. Thus, the casualties were under control, as indicated by the battle glyphs. The blue armies were still strong and could prepare for the next attack. Figure 6 bottom presents another example. The blue and the red armies encountered and fought with each other. However, the red air forces were commanded to fly through obstacles to surround the blue armies. Under this circumstance, almost all of the red units could attack the blue units, but only a small portion of the blue units could fight back. Even worse, as the blue armies were mainly composed of ground forces, the only way (bottom left direction) to escape was blocked. After suffering serious casualties, the blue nation surrendered.

Case 2: Unit Types. Figure 7 shows an example in which units of

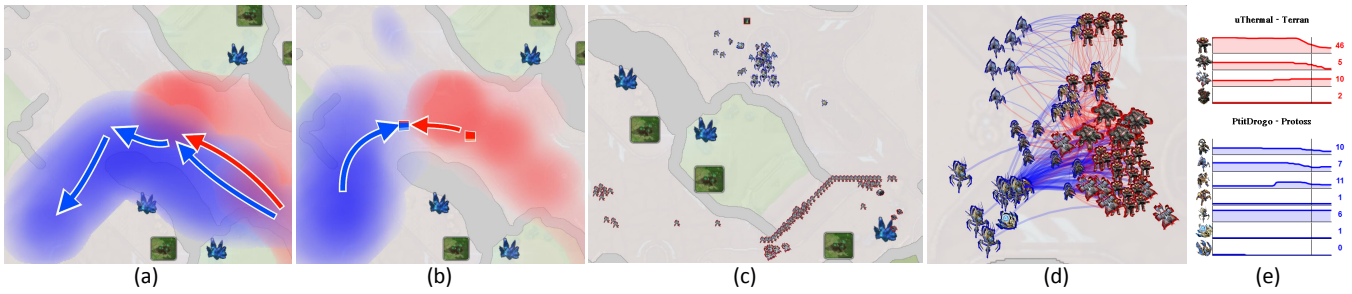

Figure 8: (a-b) The sequential key frames in a battle. Because colossi are effective at distant attack, they first moved back and then attacked behind the other units, as presented in the local view (c-d). (e) The line charts show that the number of marines rapidly decreased.

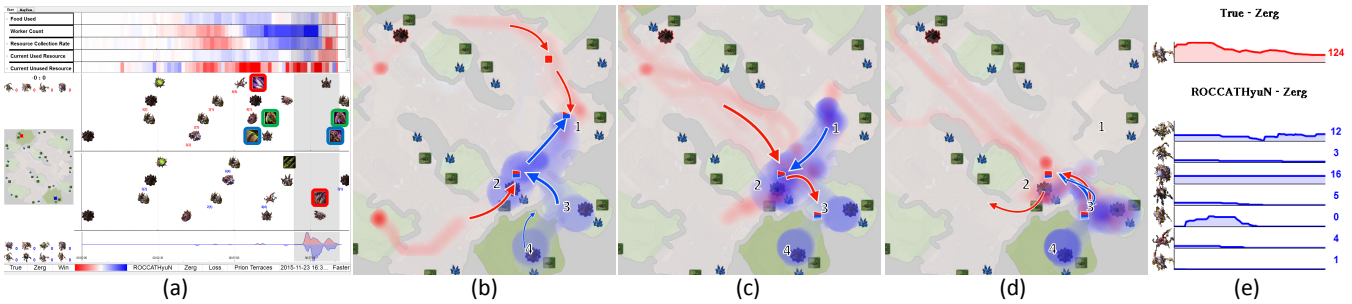

Figure 9: (a) The status view. The techniques highlighted in red, green, and blue are used to enhance speed, attack, and defense of the red units, respectively. (b-d) The key frames in the battle. A group of zerglings in red sacrificed themselves to lure the opponent's main force away from base 2. Then, the other groups of zerglings attacked the base and killed workers to slow down the resource collection of the blue nation. The blue nation surrendered in the end because they ran out of resources. (e) The line charts present the military strengths of the two nations.

a type were prevailed by units of another type. In this example, blue and red are the nations established by protoss and zerg, respectively. In the battle, blue armies first gathered and prepared for the attack. There were 18 adepts, 9 stalkers, and 1 warp prism in blue, and 15 hydralisks, 16 overlords, 15 zerglings, 2 queens, 2 ravagers, and 3 lurkers in red. Overall, the red armies were mainly composed of light armor and they could attack with high mobility. However, these red armies were prevailed by blue adepts. Therefore, when the battle occurred, 7 additional blue adepts were commanded immediately to join the battle (the line chart of adepts grows from 11 to 18). The blue armies then defeated the red vanguard and occupied the high ground. Although some of the red armies attempted to make a detour and attacked the blue armies' back, they arrived too late and lost the geographic advantage. In other words, although the blue armies were surrounded, they still defeated the enemies because they stood on high ground and the blue adepts prevailed against all red armies.

Case 3: Formation and configuration. The formation and configuration of units play important roles in a battle. Figure 8 shows an example. Blue and red are the nations established by terran and protoss, respectively. There were 80 marines, 7 siege tanks, and 8 medivacs in red, and 6 colossi, 1 sentry, 14 adepts, 13 stalkers, 6 zealots, 1 warp prism, and 1 immortal in blue. In the beginning, the red armies chased the blue armies but they did not actually fight, as indicated by the absence of battle glyphs (Figure 8 (a)). However, the blue armies fought back after they evacuated to the lower side of an obstacle (Figure 8 (b)). By zooming the map to examine the detailed view (Figure 8 (c)), users can observe that the colossi were commanded to stay back during the evacuation. Because colossi are effective at distant attack, when the blue armies were in this new formation, the colossi could take advantage of the distant attack against the enemies (Figure 8 (d)). As indicated in the line chart (Figure 8 (e)), the number of marines rapidly decreased and the red nation surrendered.

Case 4: Tactics. Both nations in this example were constructed by zerg, but the adopted strategies were different (Figure 9). The red nation trained many, but weak, units; whereas the blue nation adopted the opposite strategy – of few, but strong, units. At the early stage in the battle, there were 124 zerglings in red, 12 roaches, 13 hydralisks, and 4 ravagers in blue. Although the zerglings were weak, the red nation had developed several techniques to enhance their strengths (Figure 9 (a)). In other words, the red armies formed by such a large amount of zerglings had similar strengths compared to the blue armies formed by advanced units. Because the zerglings were fast, they were divided into three groups in the beginning. Namely, a small group disturbed the opponent's base 1 to attract blue units, a large group then attacked from the bottom, and another large group at the middle supported the attacks (Figure 9 (b-c). The strategy adopted in the red nation was to lure the opponent's main force away from base 2 by sacrificing the first group. Then, the other two groups of zerglings attacked the opponent's base 2 and killed as many workers as possible to slow down the resource collection. Because more and more relief zerglings joined the battle, the red armies got the first victory and then attempted to attack bases 3 and 4. However, the corridor to bases 3 and 4 were thin and allowed only a small amount of zerglings to pass through at a time. Thus, many zerglings were killed by the newly trained roaches (Figure 9 (d)). So far, the armies in the two nations had approximately the same strengths (Figure 9 (e)). However, since many workers in the blue nation had been killed, the resources were much fewer than that in the red nation, as indicated in the status view. Ultimately, the blue nation surrendered.

5 User Study

We conducted a user study with 20 participants to evaluate the presented visualization system. In the beginning, they received a tutorial of the presented system, Scelight, and GGTracker. Then,

they were asked to answer questions related to strategies of four games with the assistance of systems. After using the tools, they had to provide us feedback.

5.1 Compared Systems

Most users review an SC2 game by using the official replay system. It is the easiest way for them to achieve this because the replay and the gaming systems are built together, and the appearances are identical. During the review, users can pan and zoom the viewpoints to different perspectives, control the speed, and directly jump to a particular time span of a replay. Besides the official system, there are two publicly available visualization systems called Scelight and GGTracker. They provide users with an iterative interface to examine certain statistics, such as resources and military strength, over time by line charts. The goal is similar to the color-coded charts in our system (Figure 2 (a)). Besides the line charts, Scelight conveys the build order in a game, which is similar to Figure 2 (c) in our system. GGTracker also depicts the distribution of units by scatter points on a small map. Users can control the time slider and watch the map to have an overall idea of units' movements. Both the systems, however, do not focus on visualizing strategies in a battle.

Because the official replay system provides all the details of a game, we compared two sets of the systems in this study. They are *Scelight + GGTracker + Official* (SGO in short) and *Ours + Official* (OO in short). We attempt to know the behaviors of the participants when using the two sets of visualization tools to examine a game.

5.2 Participants

The participants were sought from the Internet. Their game ages in playing SC2 ranged from 0.5 to 7 years (M=4.5, SD=2.68). Because most SC2 players joined the Battle.net leagues and were ranked in this world wide organization, to obtain feedback from both beginners and experts, we chose the participants to cover as wide rank as possible. Specifically, the rank from the highest to the lowest are grandmaster (top 1000 players), master (top 4%), diamond (4-23%), platinum (23-46%), gold (46-78%), silver (78-96%), and bronze (96-100%). Except the grandmaster level, in which the players were too few, we recruited the participants and make sure at least one participant falling into each of the remaining levels.

5.3 Study Procedure

The user study was conducted in a quiet room. The participants used a desktop PC that can run the visualization systems interactively, with a 22 inch, full resolution screen. Each of them is accompanied by a nominator and was asked to think aloud during the study so that we would know what they were finding and thinking.

The participants first received a tutorial to learn the interfaces of Scelight, GGTracker, and our system. The official game replay system was not instructed because all of the participants were familiar with it. After the tutorial, we showed the participants a case study (the example in Figure 6 bottom) by using SGO and OO and allowed them to freely experience the tools until they felt that they were fluent in operating them. Next, we asked the participants to examine four game replays with the assistance of SGO and OO, respectively, and tell us why a nation beat the other in each game. Specifically, before examining the game, we asked the participants *"Why was the nation defeated? Please use the tool to discover the production, resources, techniques, buildings, and strategies used in the battles, and tell us the reason."*. To prevent order effects, we randomly and evenly partitioned the participants into two groups. The first group examined two games with the assistance of SGO and then the other two games by OO. The second group had an opposite order. We also recorded the time duration taken by the participants when the nominator considered that the questions were correctly answered. At the completion of answering all of the questions, they filled out a questionnaire to rate their perceived satisfaction to the systems.

The four selected games were played by the players at the level of grandmaster and were downloaded from the Spawning Tool. The races played in the games were zerg vs. protoss (G1), protoss vs. terran (G2), terran vs. terran (G3), and zerg vs. zerg (G4), respectively. In addition, the difficulties of answering G1, G2, G3, and G4 were simple, simple, median, and hard, respectively, which were discussed with the participants who joined the pilot study. The difficulty was determined based on the number of events, the total area of battlefields, the number of unit categories, and the difference of military strengths between two nations. In other words, we consider the problem to be easy if the game had few events, the total battlefield covered a small portion of the map, the two nations fought with few types of unit categories, and the military strengths of the two nations had significant difference. The problem was considered to be difficult in the opposite situation. Because the games were selected from a world championship, the broadcast of these games was publicly available on the Internet. We obtained the answers from reporters.

5.4 Quantitative Results

Figure 10 left shows the box and whisker plot to depict the answering time of G1-G4 by using SGO and OO, respectively. As indicated, the participants took less time to answer questions by using our system. We remind readers that the difficulties of answering questions G1-G4 were simple, simple, median, and hard, respectively. The answering time, however, was not proportional to the difficulty because the lengths of these four games were different. The difficulty here means the mental effort that participants had to exert to find the key factors of defeat and victory.

Statistically, the p-values of the answering time between SGO and OO in these four questions were 0.49, 0.55, 0.28, and 0.03, respectively. These values implied that our system was not considerably over the combination of Scelight and GGTracker if the cause of defeat or victory was clear and could be easily observed. However, our system would be helpful if the difficulty increased. Besides the difficulty, we observed that several participants tended to answer the questions even though the cause of defeat and victory was not revealed on the screen. This behavior was more frequent when using the SGO than OO because they could not obtain the answer by examining line charts and build orders, and had to switch to the official system. Consequently, when answering G1 and G2, they had a greater chance to guess the correct answers and completed the task in a very short time. In contrast, guessing the correct answer in G4 was not easy. They had to understand the strategies used on the battlefield to correctly answer the question. We also show the use-rate of visualization tools in Figure 10 middle because the participants could use the official system to seek or confirm the answers. In other words, we exclude the time that the participants spent on the official system and visualize the use-rate of tools in the study. As can be seen, by using our visualization system, the participants took less time and received greater support from our system than Scelight+GGTracker when they discovered strategies. This phenomenon can be observed in all of the questions.

5.5 Self-Rated Measures

The participants self-rated seven usability related measures at the end of the study. Figure 10 right shows the results, where the high value indicates high satisfaction. The participants preferred our system over Scelight+GGTracker in satisfaction, ease of use, learnability, mental effort, physical effort, time, and helpfulness because our system integrated the resource statistics, build order, death ThemeRiver, and a battle view for examination.

5.6 Qualitative Feedback

The feedback for the status view and build order was quite diverse. The participants at high levels tended to know the amounts of min-

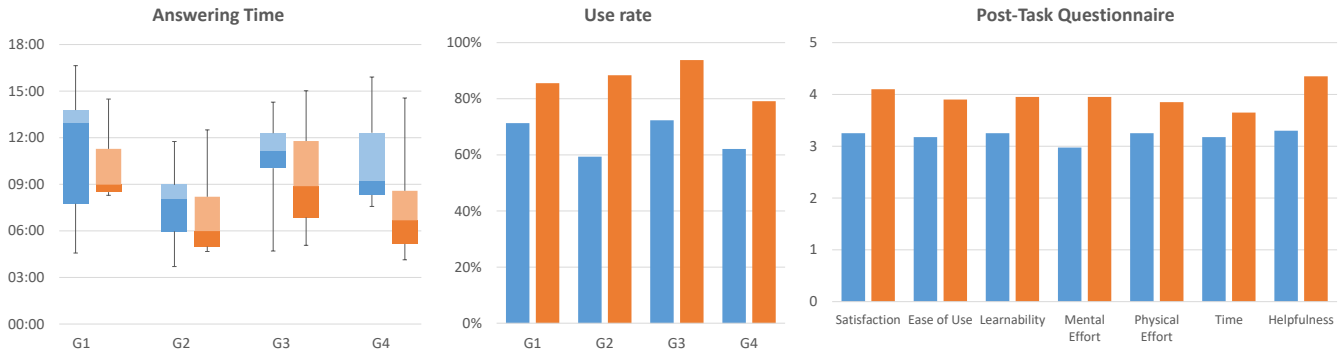

Figure 10: Quantitative results and self-rated measures in the user study. The statistics for SGO and OO are shaded in blue and orange, respectively. (Left) The time that the participants spent on answering the questions. The five-number summaries from top to bottom represent the max, 25^{th} percentile, median, 75^{th} percentile, and min, respectively. (Middle) The use rate of the visualization tools in finding the answers. (Right) The self-rated measures. The high value indicates high satisfaction.

erals and vespene gas, the types of techniques and buildings, and the numbers of units of each type over time in a game. They could understand the strategy and predict the defeat or victory of a nation according to the conveyed statistics. Therefore, although the line charts in Scelight and GGTracker are visually cluttered, the participants can still obtain the desired information after switching attention in views back and forth. When using our system, they identified that major advantages were the integration of resource statistics and build order. Because the views shared an identical time coordinate system, they could easily consider different information and build a global concept without a long sequence of operations. However, they disliked the relative statistics depicted by the color-coded charts because they were very familiar with absolute statistics in SC2 games. Hence, they suggested us to pop out a sub-window close to the cursor when users click at a time span of a color-coded chart. The sub-window may visualize a line chart and indicate the absolute value of the clicked statistic. Regarding the participants at low levels, they were not that good at obtaining the strategy behind statistics and build order. They relied heavily on our color-coded charts to understand the strengths of each nation over time and could realize the strategy only after examining details in the battle view.

The participants pointed out that knowing the configuration of armies and how they attack and defend in a battle is crucial. The nation with stronger forces might be defeated in a battle, but the cause of this result could not be observed from the status view. The participants at high levels agreed the small maps in GGTracker can ease the problem to some extent. They were able to perform a quick overview of a game and infer what happened in battles according to the scatter points on the small map and their domain knowledge. However, scatter points have an identical appearance and could not differentiate different unit types or even units and buildings, which could easily result in misunderstanding. Therefore, many participants liked the battle view provided by our system because unit types and movements were clearly depicted. They particularly liked the battle selection from the death ThemeRiver because they could quickly jump to an important battle and observe the strategies. By contrast, when watching replays using the official system, they could not identify when a crucial event occurred and had to continue retrieving the event by dragging the time slider. The participants were also grateful to our system when several events appear on the map simultaneously. P8 said *"I can watch only a region on the map by using the official replay system. Sometimes I would be too busy to switch viewpoints when there are too many events. But I could investigate the events one-by-one easily when using your system."* Despite the inconvenience, however, 20% of the participants still preferred the official replay system because they believed that only the official system could provide all details they need, such as the explored area of a nation and the health bar of a unit. They also preferred the beautiful scenes rendered by the official system.

Some participants mentioned that the curved arrows in the battle view can successfully visualize a long time span of a battle, including where an army entered and escaped from the battlefield, and how the armies were surrounded. The arrows were also particularly useful to users when a number of armies joined a battle from different locations. P7 pointed the arrows on our battle view when answering the question, stating *"The blue army first attacked the opponent's base. But their position is lower than the base and then evacuated to a high ground."* Therefore, the system can be a tool for coaches to explain strategies in a game. Some participants also pointed out that the arrows could show users a global concept of the strategies so that they did not need to build the picture by themselves according to the moving trajectory of every unit. P17 said, *"Arrows are used in some post game comments as well. [...] I liked the arrow representation. It's an intuitive way to show the movements of armies."*

5.7 Discussions

The overall feedback indicated that one advantage of OO was the integration of different data types in a view. The integration aligned the time varying resources, build order, and the death ThemeRiver at the same time coordinate system. Accordingly, the participants could compare what had been done by the players in different perspectives and then learned strategies. In addition, the death ThemeRiver indicated the time span that each battle occurred and allowed the participants to observe the influence of the battle in the subsequent developments. Since the participants could switch the view to the battlefield, they also could discover offensive and defensive strategies, such as formation, prevailing between units, and ground height, in the battle. For example, they were able to observe the abstracted arrows to know how the armies were surrounded from a global view; and to observe the formation of units to learn how experts took advantages of geographic features and properties of different unit types from a local view. In contrast, Scelight and GGTracker showed the time varying statistics by line charts independently. The participant would have no idea the relations between resources, buildings, and techniques, unless they checked the time span of each data type and built the relations by themselves. In addition, these two visualization tools did not contain the battle view. When the participants were about to know why a nation with stronger forces surrendered, they had to switch the view to the official system. Since the participants were unaware of the time of the critical event, they still had to spend time on retrieving it when using the official system. By summing up, we conclude that OO was superior to SGO.

6 Conclusions and Future Works

We have presented a visualization system for users to study SC2 games. By examining statistics, build order, and battles interactively, users can learn strategies and experiences from games played by professionals. The insights identified in the visualization results and the feedback obtained from the conducted user study confirm the feasibility of our system. Although the presented system is designed mainly for SC2 games, it has the potential to visualize other RTS games sold in the market because the games share many properties in common. Therefore, we will improve our prototype program to a product level system and share it on GitHub with users worldwide.

Participants in the conducted user study pointed out several shortcomings in our visualization system, including visual quality and lack of some details in a game. They also pointed out that they had to tune the parameters γ and ρ to control the level of details of armies' movements and the arrow size, respectively. As these two parameters are related to the zoom level of the battle view, they expected the parameters to be automatically determined. In addition, although we have strived to prevent occlusions of the curved arrows on a map, occlusions sometimes are inevitable if the arrows are too large or bent to avoid passing through obstacles. Figure 9 (d) shows an example of this. We plan to solve the above-mentioned problems and improve the usability of our system accordingly in future. Besides, our system is presented to visualize one game at a time. Users are not able to observe multiple games simultaneously by using it and learn game behaviors from a certain player. To extend the presented system, the goal could be detecting similar patterns with respect to production, techniques, and strategies from a set of games and highlighting the patterns. We consider the extension as our future works.

Acknowledgments

We thank the anonymous reviewers for their constructive comments. We are grateful to Jimmy Hsieh for narrating the accompanying video, to Ching-Hsiang Kang for the discussions related to player requirements, and to all the users who participated in the user study. This work was supported in part by the Ministry of Science and Technology, Taiwan (104-2221-E-009 -041 -MY3 and 105-2221-E-009 -135 -MY3).

References

[1] T. Avontuur, P. Spronck, and M. V. Zaanen. Player skill modeling in starcraft II. In *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2013.

[2] C. Ballinger, S. Louis, and S. Liu. Coevolving robust build-order iterative lists for real-time strategy games. *IEEE Transactions on Computational Intelligence and AI in Games*, 8(4):363–376, Dec 2016.

[3] A. Belicza. Scelights, 2013.

[4] A. R. Gagné, S. El-Nasr, Magy, and C. D. Shaw. Analysis of telemetry data from a real-time strategy game. *Computers in Entertainment*, 10(1):2, 2012.

[5] S. Havre, B. Hetzler, and L. Nowell. Themeriver: Visualizing theme changes over time. In *INFOVIS '00: Proceedings of the IEEE Symposium on Information Vizualization*, p. 115, 2000.

[6] J. L. Hsieh and C. T. Sun. Building a player strategy model by analyzing replays of real-time strategy games. In *International Joint Conference on Neural Networks*, pp. 3106–3111, 2008.

[7] D. Joerg. Ggtracker, 2012.

[8] Y.-E. Liu, E. Andersen, R. Snider, S. Cooper, and Z. Popovi. Feature-based projections for effective playtrace analysis. In *International Conference on Foundations of Digital Games*, 2011.

[9] C. Low-Kam, C. Raissi, M. Kaytoue, and J. Pei. Mining statistically significant sequential patterns. In *International Conference on Data Mining*, 2013.

[10] K. Madsen, H. B. Nielsen, and O. Tingleff. Optimization with constraints, 2nd ed., 2004.

[11] B. Medler, M. John, and J. Lane. Data cracker. In *SIGCHI Conference on Human Factors in Computing Systems*, pp. 2365–2374, 2011.

[12] S. Ontañón, K. Mishra, N. Sugandh, and A. Ram. Case-based planning and execution for real-time strategy games. In *International Conference on Case-Based Reasoning: Case-Based Reasoning Research and Development*, pp. 164–178, 2007.

[13] S. Ontañon, G. Synnaeve, A. Uriarte, F. Richoux, D. Churchill, and M. Preuss. A Survey of Real-Time Strategy Game AI Research and Competition in StarCraft. *IEEE Transactions on Computational Intelligence and AI in games*, 5(4):1–19, 2013.

[14] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, 2 ed., 2003.

[15] G. Synnaeve and P. Bessière. A Dataset for StarCraft AI & an Example of Armies Clustering. In *Artificial Intelligence in Adversarial Real-Time Games 2012*, pp. pp 25–30, Oct 2012.

[16] G. Synnaeve and P. Bessière. Multiscale bayesian modeling for rts games: An application to starcraft ai. *IEEE Transactions on Computational Intelligence and AI in Games*, 8(4):338–350, Dec 2016.

[17] R. Thawonmas and K. Iizuka. Visualization of online-game players based on their action behaviors. *International Journal of Computer Games Technology*, 2008:5, 2008.

[18] A. Uriarte and S. Ontañón. Combat models for rts games. *IEEE Transactions on Computational Intelligence and AI in Games*, PP(99):1–1, 2017.

[19] G. Wallner and S. Kriglstein. A spatiotemporal visualization approach for the analysis of gameplay data. In *SIGCHI Conference on Human Factors in Computing Systems*, pp. 1115–1124, 2012.

[20] G. Wallner and S. Kriglstein. Visualization-based analysis of gameplay data a review of literature. *Entertainment Computing*, 4(3):143155, 2013.

[21] G. Wallner and S. Kriglstein. Plato: A visual analytics system for gameplay data. *Computers & Graphics*, 38:341356, 2014.

[22] B. G. Weber and M. Mateas. A data mining approach to strategy prediction. pp. 140–147, 2010.

[23] S. Wender, A. Cordier, and I. Watson. Building a trace-based system for real-time strategy game traces. In *EXperience reuse: Provenance, Process-ORientation and Traces*.