Visualizing Student Histories Using Clustering and Composition

David Trimm, Penny Rheingans, *Member, IEEE*, and Marie desJardins

Abstract—While intuitive time-series visualizations exist for common datasets, student course history data is difficult to represent using traditional visualization techniques due its concurrent nature. A visual composition process is developed and applied to reveal trends across various groupings. By working closely with educators, analytic strategies and techniques are developed to leverage the visualization composition to reveal unknown trends in the data. Furthermore, clustering algorithms are developed to group common course-grade histories for further analysis. Lastly, variations of the composition process are implemented to reveal subtle differences in the underlying data. These analytic tools and techniques enabled educators to confirm expected trends and to discover new ones.

Index Terms—Clustering, aggregate visualization, student performance analysis, visualization composition.

1 Introduction

Time-series data has always been challenging to visualize and analyze across a wide range of applications. Generic visualization solutions exist for many types of time-series data types, most notably xy-scatter plots and trend lines for scalar values over time [1]. These widely-used visualizations are highly intuitive to understand trends and identify anomalies. However, time-series data capturing set-based elements is difficult to analyze and has not been adequately addressed. Our focus for this paper is visualizing student course history data—a widely collected dataset across all levels of academics, that when analyzed, can yield significant insights into student performance.

We use the following definitions for the *student course history* domain. For discrete time durations, referred to as *semesters*, a student attends multiple courses. Over the course of a student's academic career, multiple semesters are completed with increased expectations as the student progresses towards a degree. *Courses* have intrinsic characteristics, such as *academic discipline* (e.g., history or mathematics), *prerequisites*, and a level designation denoted by a *course number*. When associated with a student, extrinsic *attributes* become associated with a course—final grade, semester taken, instructor, etc. Similarly, students have intrinsic associations, referred to as demographic data, and attributes that change over time, most notably accumulated GPA. This model can be generalized for other applications. For example, dietary history for a weight loss program or daily personal spending binned by category could both be modeled similarly.

Analyzing this data is challenging due to the concurrent nature of courses in the student history. For a slice in time, a student attends multiple courses, breaking the definition for a function (i.e., a single time maps to multiple course values) and eliminating most temporal visualization approaches. The problem is further complicated by both the intrinsic and extrinsic characteristics of the courses and students. However, if the temporal and concurrent aspects could be accurately represented, analytical conclusions derived from the aggregate student trends could then be used to improve academic programs.

The approach leverages visual composition techniques to depict trends in student course history data. First, students are grouped by the grades that they receive for specific courses—most notably, the gateway and math courses. *Gateway courses* expose starting students to the core concepts of the major and determine a student's aptitude for the coursework. After the groupings are created, the visual composition techniques are applied to identify overall performance, trends across time, and aggregate semester grades through a *small multiples* [13] approach. In addition to examining course-grade dependencies, students are also grouped by demographic categories to identify related correlations. Based on feedback from academic experts, the approach is modified to find courses that are critical to later success and the composition technique is further modified to depict minute differences in student groups. The results successfully visualize expected trends and discover unexpected relationships in student performance.

- *David Trimm, University of Maryland, Baltimore County (UMBC), e-mail: dave.trimm@gmail.com.*
- *Penny Rheingans, UMBC, e-mail: rheingan@cs.umbc.edu.*
- *Marie desJardins, UMBC, e-mail: mariedj@cs.umbc.edu.*

Manuscript received 31 March 2012; accepted 1 August 2012; posted online 14 October 2012; mailed on 5 October 2012.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

2 Application

When analyzed in aggregate, student course history data can provide a deep understanding of the many factors influencing an individual student's performance. In addition to identifying student-dependent variables, the analysis can aid in recognizing deficits in existing academic programs and potential methods for improving the curriculum. In both cases, examining individual student information in isolation would not yield equivalent results—only when the data is analyzed in aggregate do general trends become apparent.

At the individual student level, analyzing trends in past student performance can aid in predicting future performance. For example, analyzing and correlating a student's high school GPA and SAT score with the course plan that they pursue can determine which factors play the most significant role. Alternatively, characterizing unsuccessful student strategies can identify existing students on the same downward trajectory. These students can then be counseled towards more productive strategies, resulting in a successful outcome for both the student and the academic institution.

Whereas improving the individual student's academic career is a local optimization, the conclusions from analyzing student data can also help to refine and shape the curriculum, leading to global optimizations. A curriculum is composed of a recommended course plan including prerequisites and a course-to-topic coverage model that ensures graduates are well rounded. By determining dependencies among prerequisites and other coursework, academic program developers can improve the existing curriculum. Improvements may include requiring new prerequisites, modifying the topic coverage for specific courses, or determining threshold grades for students to progress to more advanced courses. By enacting global optimizations, an academic institute can increase retention and graduation rates, leading to a more successful program for both students and the institution. Course history data can also help determine new policies and procedures. For example, the effect that course load has on student performance could be used to determine new restrictions on the maximum number of courses that a student could take per semester.

In summary, the approach needs to reveal aggregate student-level performance and curriculum-driven features. Such a technique would then address the following questions:

- Which courses are most critical for later success?

- Are there places in the curriculum where students struggle to progress smoothly?
- Which demographic features play a role in student success?

3 Related Work

The most similar projects to this research are visualizations applied to similar data. Traditional techniques were the most prevalent, and the analytical goals (i.e., improving student performance) directly aligned with the goals and objectives. Wortman et al. [16] developed an application to interactively explore the first three courses in computer science using traditional visualization techniques. Edwards et al. [4] provided students with feedback based on data collected from electronic submission systems. Using traditional visualization techniques, student feedback was provided that gaged their progress against fellow students. CourseViz [8] visualized web-based data to identify trends in behavior and identify remote students in need. DynMap [10] visualized the evolution of computer security students through particular subjects using concept maps. Two studies were performed to confirm that the characteristics were appropriate and understandable through the visualization system. While these efforts gave additional insight into student performance, most of them relied on traditional techniques that only focused on a narrow window in time.

For visual composition of aggregate time series data, several other efforts drive toward similar goals but use different techniques. Muigg et al. [9] apply tensor fields during rasterization to scalably represent time-series lines. While their technique is applied to parallel coordinates, it could easily be applied to our representation of student course history data. Other parallel coordinates efforts are also similar. For instance, Heinrich and Weiskopf [7] aggregate continuous scatter plots to mathematically model the density of parallel coordinates data. Their efforts derive a similar aggregate composition that could benefit from the color scheme we use to depict application-specific data.

4 Approach

Our approach to student course history analysis is to group student trajectories by their course grade combinations across their academic careers. These related groupings are then visualized using compositions that show the characteristics of the group's structure, variances in course history, and attributes across time (e.g. accumulated GPA, high school GPA, etc.). In order to apply our composition techniques, each student's career history is represented by a simple, but effective, two-dimensional trajectory. By first partitioning the related students into distinct partitions, a cohesive composition can be created of students who share similar course histories. Alternatively, compositions can be formed from demographic slices—this enables analysis of external or intrinsic factors which may influence a student's performance, including those attributes which will predict future performance. Early reviews by academic experts led us to develop clustering techniques to find similar course-grade histories and to develop fine-grained methods to detect differences in semester GPAs.

4.1 Trajectory Representation

To successfully use the composition process, each student's course history is represented as a two-dimensional trajectory. Trajectory representations are complicated by the fact that a single semester contains multiple courses. Discrete values for each semester are, therefore, difficult to create. If each course is individually examined, comparisons among different students who may only partially overlap in a semester are difficult to make—for instance, what ordering would be used when only some courses overlap? To solve this problem, the overall average of the course numbers is used to represent the y-axis value for a semester. At our institution, most students incrementally take higher numbered courses over their academic career, especially within their declared major. To maintain the representation's temporal characteristics, the x-axis is used to represent the student's semester index. Figure 1 provides a graphical depiction of the spatial trajectory for a simplified student course history. In the figure, the student's trajectory slopes upward from left-to-right as higher-level courses are taken.

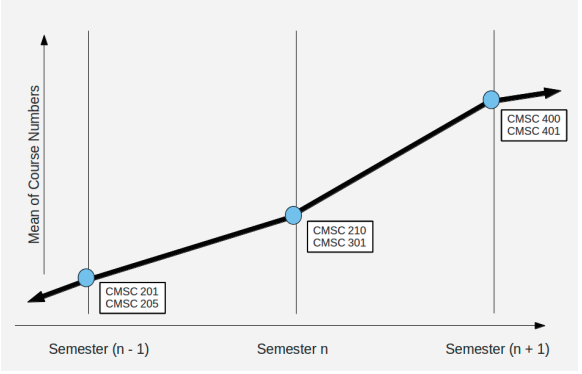

Fig. 1. Two-dimensional representation for student course history. The x-axis corresponds to the semester index while the y-axis represents the mean of the course numbers for a semester.

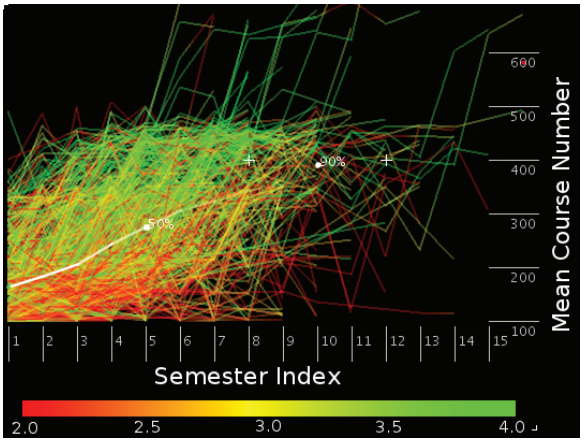

Fig. 2. Two-dimensional representation of historical student course data. Student trajectories are colored by semester GPA.

The most critical assumption for the spatial representation is that increasing course numbers correlate strongly with student progression and advancement. At our institution, this assumption holds true—a careful examination of the recommended curriculum shows that the course numbers increase monotonically across a student's career. However, for universities where this is not the case, a different metric would need to be developed corresponding to knowledge increments. For example, the aggregate number of prerequisites could be used to score each course with a numerical value. Alternatively, the core concepts for the major could be mapped to each course in the curriculum resulting in a fine grained metric. In our results, the areas of higher divergence in our results tended to occur at the end of student careers due, in part, to specialization at the 400-level series. However, for the introductory math, science, and gateway courses through the mid-level courses, the data correlated well across the groupings.

Courses not directly related to a student's major were removed from the analysis. In order to keep the spatial representation clear, only courses directly related to the curriculum—i.e., computer science, computer engineering, mathematics, and sciences—were used.

4.2 Composition Process

Unfortunately, attempting to visualize the trends and characteristics of many students at once is difficult even when using a simple representation, as described in the last section. Figure 2 represents all of the computer science students in the sample. The figure uses the previously described two-dimensional representation and is rendered by coloring each line segment with the semester GPA. In the non-composited figure, it is possible to tell that students with flat trajectories tended to have lower grades—i.e., students who remained in low-

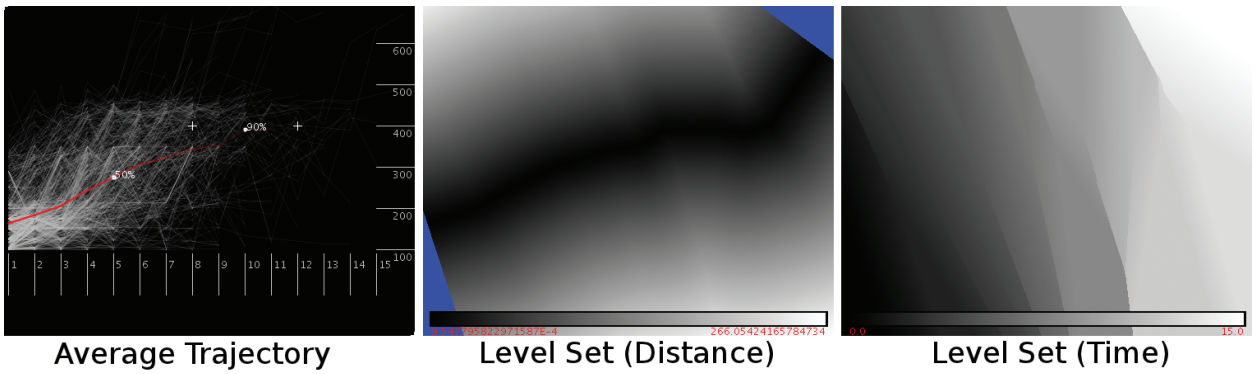

Fig. 3. Composition phases. From left-to-right, the figure shows the average trajectory calculation, the level set distance mapping, and the level set time parameter mapping.

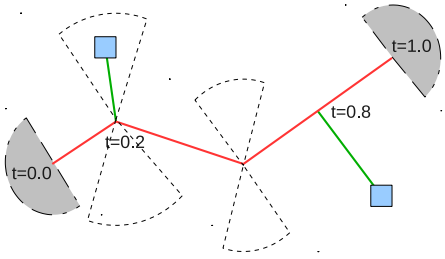

Fig. 4. Parametric trajectory transformation diagram. The modified level set algorithm determines how pixels in the final rendering are mapped to parametric values along the average trajectory.

numbered courses had red and yellow trajectories. However, meaningful comparisons are difficult at best.

Unless otherwise noted, a green-yellow-red color scale will be used for GPA on the 4.0 to 2.0 bounded interval. GPAs below 2.0 will map to red. These colors provide an intuitive mapping [2]—students doing exceptionally well are shown as green while those students clearly in trouble are red. Yellow as the intermediary provides an expected transition between red and green. It should be noted that this color scheme is problematic for color blind individuals due to the use of red and green. However, for this application, the use of "stop-light chart" colors provides an extremely intuitive representation to understanding the data—in accordance with Silva et al. [12], "...the most striking features of the image reflect the most important features of the data." Furthermore, the approach is flexible and can use other color schemes to represent the trends in the student data—in the results section, an alternate colorscale is used to depict the data.

Figure 2 approximates a traditional line graph and, to some degree, a parallel coordinates view with several notable distinctions. First, the axes do not represent different dimensions but instead correspond to discrete times (i.e., more similar to a line graph). Second, each element in the chart (i.e., a single student) does not span every coordinate axis—instead, the student's trajectory ends when they either graduate or otherwise end their academic careers. Density is the most observable characteristic of the straightforward approach; however, this comes at the expense of multiple course history lines overwriting one another. While opacity could be used to preserve the GPA values, color blending would result in a poor depiction of the results since only the colors are merged, not the underlying data values. Furthermore, local deviations are not represented in a meaningful way. These deficiencies led us to develop a composition approach for aggregate trajectories.

The trajectory composition structure needs to (1) show the overall *spatial characteristics* of the trajectories (e.g., course number progression) and (2) clearly denote the trend in the *student-related attribute* (e.g., semester GPA). *Spatial characteristics* enable the viewer to see the general shape, density, compactness, and spatial variances of the trajectory set. For example, spatial variances along the vertical axis indicate differences in the course level for that semester. Furthermore, spatial density corresponds to the distribution of trajectories—e.g., tightly coupled together or generally spread out. Likewise, *student-related attributes* enable the viewer to understand how an attribute changes over time. For the application, the requirement is to depict both the mean and variance for the semester GPA.

To satisfy these goals, we chose to use the set's average trajectory as a common frame of reference for the composition. The trajectory averaging algorithm is to simply calculate the average value for all student trajectories in the group at each time increment (i.e., semester index). Figure 3 (leftmost tile) shows the average trajectory for all of the students in our data. In the figure, the red line indicates the average for the underlying trajectories shown in white. To fill the composition, each "contributing trajectory" needs to be transformed to the common reference frame (i.e., average trajectory) to aggregate its contribution. The transformation then ensures that the contributing trajectory accurately influences the appropriate pixels in the composition.

Level Sets [11] model the expansion of a two-dimensional boundary over time from an initial starting condition. The starting condition represents the boundary at the initial time (i.e., t_0). By iteratively expanding the boundary, the Level Set algorithm determines how the boundary propagates and when it will reach a particular location. For a boundary that expands at a constant rate, the time to reach a point is equivalent to the distance to the initial boundary. Figure 3 (middle tile) depicts an example of the Level Set algorithm when the average trajectory is used as the starting condition. The brightness represents the boundary expansion (and distance from average trajectory) over time with black equaling t_0. In addition to storing the distance from the average trajectory, the level set algorithm is modified to retain the parametric time parameter from the average trajectory—recall that this corresponds to the semester index.

Once the modified Level Set algorithm determines the parametric and distance values for each pixel, the contributing trajectories are transformed and composited together for the final rendering. The final objective for this stage is to accumulate the individual trajectory contributions for each pixel in the composition; the contributions are captured as a list of trajectory attribute values for the corresponding trajectories that influence a particular pixel. The per-pixel list of attribute values is then used to select the pixel's hue and is described in more detail in the next chapter. For each pixel, we need to determine which contributing trajectories influence its appearance. To accomplish this, the corresponding spatial location is determined on each contributing trajectory on a pixel-by-pixel basis—recall that each pixel has a parametric value associated with it as a result of the Level Set. The parametric value is then used to interpolate along the contributing trajectory to find its corresponding spatial location. This spatial location is then compared with the average trajectory to determine the distance and relative position (above or below the average) in composition space. To achieve the visual envelope appearance, the contributing trajectory influences all distance values less than or equal to its

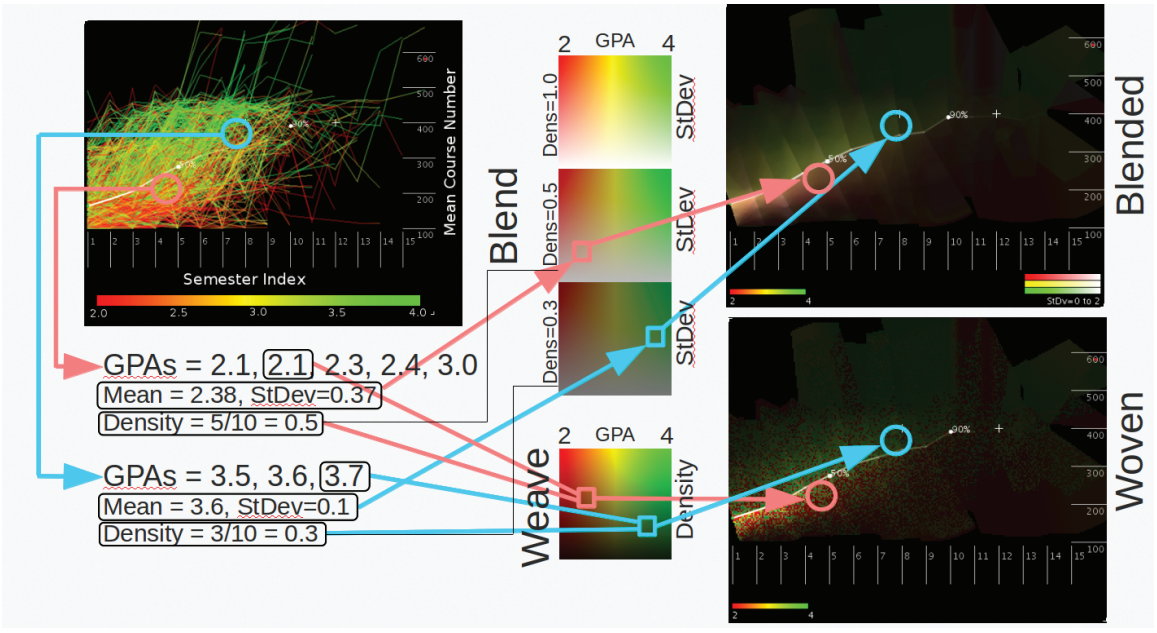

Fig. 5. Color composition. From the trajectory data (upper left), the associated GPAs are listed for each pixel (lower left). When blending is used for composition, the average, standard deviation, and density (number of trajectories contributing) are used to select the appropriate blending color (upper middle). Alternatively, for a woven composition, a random GPA is chosen, and when combined with the density, selects the appropriate color.

distance from the average trajectory for that parametric value.

4.3 Visualization

The final phase of the composition process is to assign color values to each pixel in the final rendering. Several goals need to be satisfied by the chosen scheme. First, the spatial density of the contributing trajectories needs to be effectively depicted, providing the viewer with a clear understanding of the spatial structure. Second, the trajectory attribute values should be discernible by the viewer in their corresponding spatial location—by properly showing this value, the attribute trends along the trajectory can be intuitively understood. Finally, if the contributing trajectory attributes have a wide variance, that should be depicted as well. In general, pixel brightness provides an intuitive representation for areas of high density, and pixel hue is a common choice for attribute values. For this specific application, density should indicate the most common trajectory taken by students within the course-grade set, and hue should convey information about GPA: both the average value across students and variations within the group.

We chose to use two separate coloring approaches. The first method uses a color blending approach [3] to account for the density, mean, and standard deviation, and the other method weaves colors [5] together to show the attribute value from a randomly selected trajectory. To render the color-blended composition, the mean, standard deviation, and density value feed into the Hue, Saturation, and Brightness (HSB) of the color models, respectively. In order to display intuitive colors for our application, we chose to use a Green-Yellow-Red colorscale to represent GPA values on the 4.0 to 2.0 scale, respectively. By modulating the saturation by the standard deviation, areas where most of the contributing trajectories have similar attribute values deepen in saturation. Areas where the contributing trajectories disagree or have a wide variance have low saturation and appear washed-out (i.e., white or gray). Finally, density corresponds directly with the brightness component, and, against a black background, provides an opaque representation where the maximum density is reached.

In Figure 5, the color blending step is shown in the upper middle of the figure. The first example pixel has more trajectories contributing to the composition and therefore results in a brighter color since the density is higher. The first pixel also has a lower mean GPA value and a higher variance, resulting in a lower saturation on the lower end

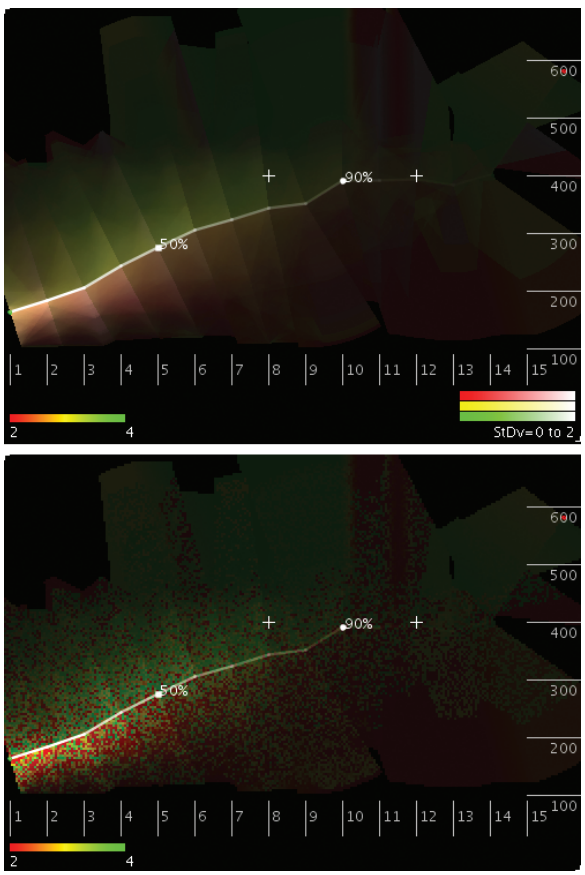

Fig. 6. Blended and woven compositions for all computer science students. Note the average upward trend and separation of higher grades (green) versus lower grades (red).

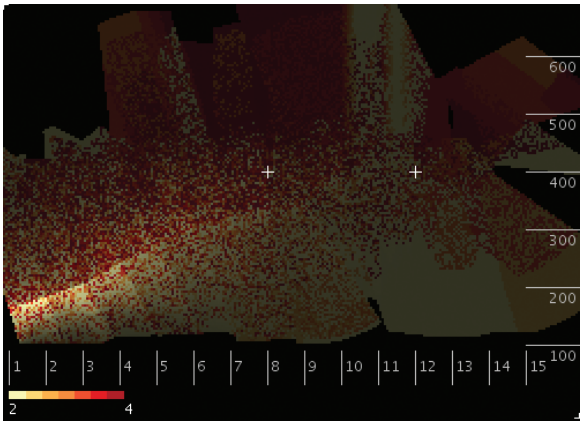

Fig. 7. Woven version using a sequential Brewer colorscale. While simpler, the reduced scale does not provide the same distinguishing characteristics of the previous figure.

of the color scale. On the other hand, the second example pixel has fewer trajectories (less density) and therefore uses a darker color value that, when placed against black, provides the intuitive appearance of less density. The second pixel has a higher mean GPA value but lower variability, resulting in a higher color value that is more saturated.

The weaving method also uses the same HSB color model, but does not modulate the saturation by the standard deviation. Instead, the algorithm randomly chooses a trajectory's attribute value and selects the appropriate hue component for the weaving cell. Density is still used with the brightness component to correctly convey the number of students and therefore the group's overall structure. Weaving provides the viewer with an exact value of a single data point in that region; deviation must be inferred by looking at neighboring point samples. The lower portion of Figure 5 shows a simple depiction for how the weaving color is chosen. First, a random value is chosen from each pixel list (lower left). This value selects the weaving color, and the density determines the brightness (lower middle).

5 Results

The approach is used to explore historical course data for undergraduate computer science and computer engineering students. The following case study provides real-world results of how trajectory composition can aid in understanding large trajectory sets, especially when they can be spatially represented and grouped into natural bins. Figure 6 composites all 1,456 computer science students together into a single rendering using the composition process. Both a blended approach (top) and a woven version (bottom) are shown in the figure. Several features immediately stand out. The average trajectory (the white line) has a upward slope representing that, in general, most students successfully progressed to more advanced classes over time. Ninety percent of students were finished by the tenth semester, while fifty percent never made it past the fifth semester—possibly graduating if they were transfer students. In regions below the average trajectory, the composition is tinged red, indicating that the students had lower grades (i.e., closer to Cs or 2.0s) than those above the line (shown in shades of green). There were significant variations in the grades, as shown by the washed-out appearance closest to the average trajectory. The variations can also be inferred from the woven version on the bottom—there are significantly more red patches below the trajectory. However, these patches also occur above as well, indicating a range (i.e., variation) of differing semester grades. The remainder of the results use a small multiples [13] approach to compare and contrast course-grade bins and specific demographics groups—small multiples is an approach where different combinations of the overall dataset are depicted in a grid to understand similarities and identify differences.

Figure 6 contrasts the strengths and weaknesses of each coloring scheme. The blending approach summarizes the trends through both hue and saturation. The advantage is that large amounts of data can be shown in an intuitive fashion. On the other hand, the woven coloring method forces the viewer to gauge the variability and general value for a particular region of the composition. However, since saturation is held constant, the viewer can see the actual values for a randomly sampled region of the composition. Because the weave size is relatively small, the viewer can estimate the variability across students.

Figures 2 and 6 both depict all of the students within our data set. However, the structure of the overall group becomes much clearer when composited together. For example, while some density information can be derived from the number of lines in Figure 2, it is difficult to determine the central trajectory for data. GPA information is also much more difficult to infer—especially what the average value is and where it significantly deviates. On the other hand, the composition in Figure 6 clearly shows the deviations towards the beginning of the students' careers and the overall trend throughout the student trajectories. As an alternate color scheme, Figure 7 shows the same composition but using a sequential red-orange-yellow Brewer colorscale. Because the colorscale contains only discrete color indices, the woven color model is a more natural fit than the blended model.

The computer science gateway courses, CMSC 201 (Computer Science I) and CMSC 202 (Computer Science II), exhibit the expected trajectories for the A students (Figure 8, column one)—specifically, successful advancement towards higher-numbered courses. However, students who received a B in CMSC 201 (row one, column two) struggled, relative to those receiving a B in CMSC 202 (row two, column two)—note the leveling off effect towards the eighth semester for CMSC 201; a B in CMSC 202 had little effect on the trajectory but did affect grades (i.e., more yellow). This indicates the relative importance of excelling at the concepts taught in CMSC 201. This is further emphasized by the relative performance of C students for CMSC 201 versus CMSC 202—namely, the dropout rate (50th percentile marking, although both occur early) and the ending location. The C CMSC 201 trend line (row one, column three) indicates higher performance after an initial struggle (flat initial line). A careful examination of the underlying data reveals that the increased slope was due to students taking high-level math courses (e.g., MATH 381 (Linear Methods in Operations Research), STAT 451 (Intro to Probability Theory), STAT 454 (Applied Statistics)). Since the computer science program requires a B or better in both CMSC 201 and 202 and doesn't require any of these courses, these students most likely switched to another major. The last column shows the students who took the courses multiple times before completing with an acceptable grade. CMSC 202 had 126 students who repeated the course. This group's perseverance is notable—out of all of the course-grade combinations for the gateways, these remained in school the longest but in many cases did not progress to the 400-series courses necessary for graduation.

As stated earlier, one objective for visualizing aggregate student histories is to identify critical points in the curriculum that determine a student's success. For the gateway courses, both the students receiving a B in CMSC 201 and those completing CMSC 202 after multiple attempts should have been able to successfully complete their degree requirements. However, the trajectories tell a different story. From these results, in-depth analysis should be performed to determine which skills are not adequately addressed for these students. Then, the curriculum and appropriate classes/class topics should be modified to overcome these deficiencies. Alternatively, students falling into these categories could be offered additional support to supplement their knowledge and skill sets.

The first three math courses, MATH 150 (Precalculus), MATH 151 (Calculus I), and MATH 152 (Calculus II), rendered in Figure 9 (rows one through three, respectively) show similar trends for students receiving Bs or Cs. MATH 150 is technically not required for computer science; students enrolled in this course were likely addressing gaps in their high school programs. However, students not receiving an A for this course showed the worst trends for all of the groups—i.e., very shallow progression, an earlier 50th percentile for students leaving the program, and lower semester GPAs, as depicted by the red coloring. MATH 150 B and C students also had relatively narrow densities, indi-

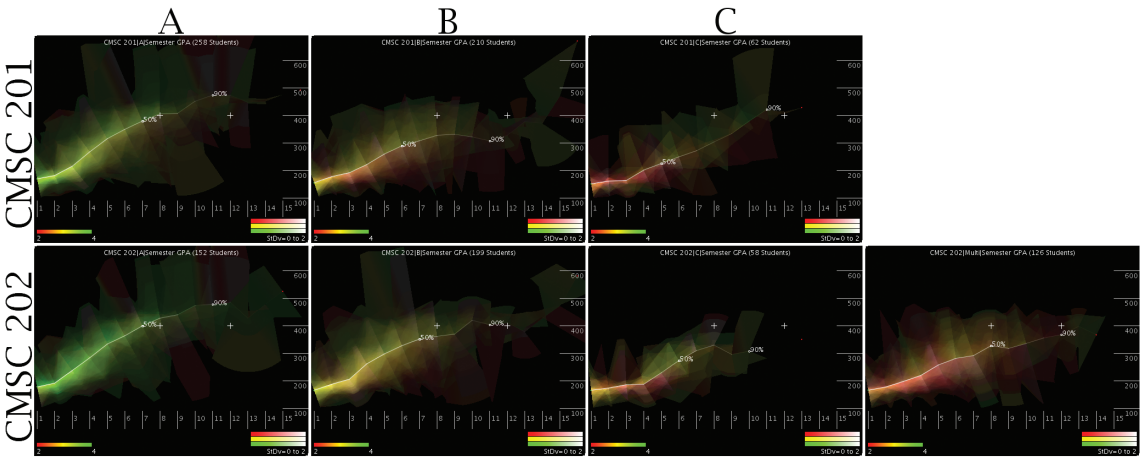

Fig. 8. Computer science gateway course trends. Small multiples shown for each gateway course-grade combination. As expected, success in foundational courses plays a significant role in future success.

cating that the students were restricted to very similar courses for their (short) computer science careers. The red areas are most prevalent for C students in either MATH 150 or MATH 151, indicating the correlation between mathematical ability and computer science success. The trend is not as apparent for other math courses (not shown), possibly revealing independence between the concepts and computer science.

Students receiving an A for the required math courses (MATH 151 and 152, rows two and three, column one in Figure 9) had a much stronger tendency for higher semesters GPAs than those in either the B or C categories. This held true regardless of which side the student fell on from the average trajectory. Note the green areas occurring on both sides of the average trajectory for the A math students as compared to the other grades—regardless of the student's trajectory, they retained high semester GPAs throughout their academic careers.

To compare high school GPA partitions with SAT Math performance (Figure 10), the students were first binned in arbitrary increments of 75 for their SAT Math scores (e.g., 800 to 725, 725 to 650). Next, the students were sorted by their high school GPA. From the sorted list, partitions were calculated to match the number of students in the SAT Math bins. For example, if there were 100 students in the highest SAT Math bin then students from 1 to 100 in the sorted GPA list were chosen for comparison.

The results between the high school GPA partitions and the corresponding SAT Math bins did not show significant differences—especially in the three highest groups (Figure 10, columns one through three). For example, the top SAT Math bin (i.e., 800 to 725) compared very similarly to the highest high school GPA bin (i.e., 5.0 to 4.21). The SAT Math results did show a higher variance in semester GPAs as shown by the somewhat washed-out appearance in the composition. However, there was only a slight increase in the standard deviation. The fourth column does show a degree of variation between the separate bins. In particular, the high school GPA group has a steady progression towards higher-numbered courses, indicating some amount of success despite the lower semester grades. The equivalent SAT Math grouping shows a leveling off around the fifth semester followed by a sharp increase in the eighth. Other than this last example, these results are contrary to conventional wisdom—high school GPA does not appear to be significantly more important than SAT Math scores.

5.1 Performance

Once the trajectories are chosen, the rendering time for each composition depends on four phases: calculating the average trajectory, performing the level set algorithm, accumulating each trajectory, and, finally, determining the statistical values and applying the appropriate color. The runtime for each phase is as follows (T is trajectories, P is total pixels in composition, S is the maximum number of semesters):

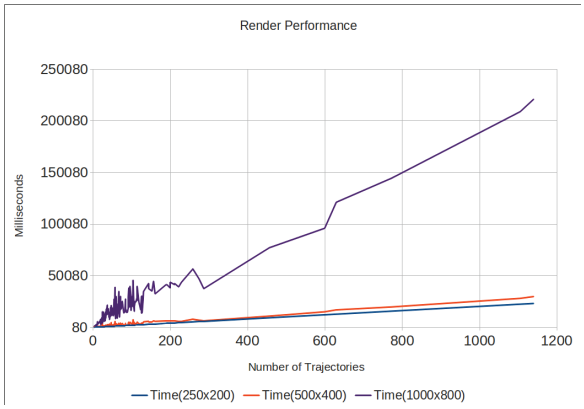

Fig. 11. Render performance for chosen images sizes.

- Average Trajectory: T x S
- Level Set Algorithm: P x O(log(P))
- Trajectory Accumulation: T x Omega(P)
- Statistics: Omega(T) x P

The number of semesters is constant and small for this application—therefore, the runtime is dependent on the number of trajectories and pixels. For the level set algorithm, we used the java TreeSet algorithm to return the next closest pixel for the boundary. The java TreeSet implementation runs in logarithmic time. The accumulation algorithm has the longest runtime overall—the worst case scenario is that a single trajectory contributes to every pixel. However, that is a rare case since most trajectories regress to the average. To achieve the runtime above, all of the parametric values were sorted prior to accumulation. By keeping the values in order, each trajectory could be interpolated in linear time at a cost of all the lookups being stored and every pixel revisited for each trajectory. To further speed up the process, this section was multi-threaded across each trajectory with minimal locking as the pixels were revisited.

Figure 11 shows the empirical values for three different composition sizes: 250x200, 500x400, 1000x800. A high-performance desktop CPU was used to perform the tests with the following specifications: Intel i7-2700K CPU, 16GB RAM, Linux 64bit OS, and the OpenJDK Java 1.7.0 environment. In the figure, the X-axis represents the number of trajectories and the Y-axis is the measured runtime in milliseconds. For an interactive application, the smaller sizes could be

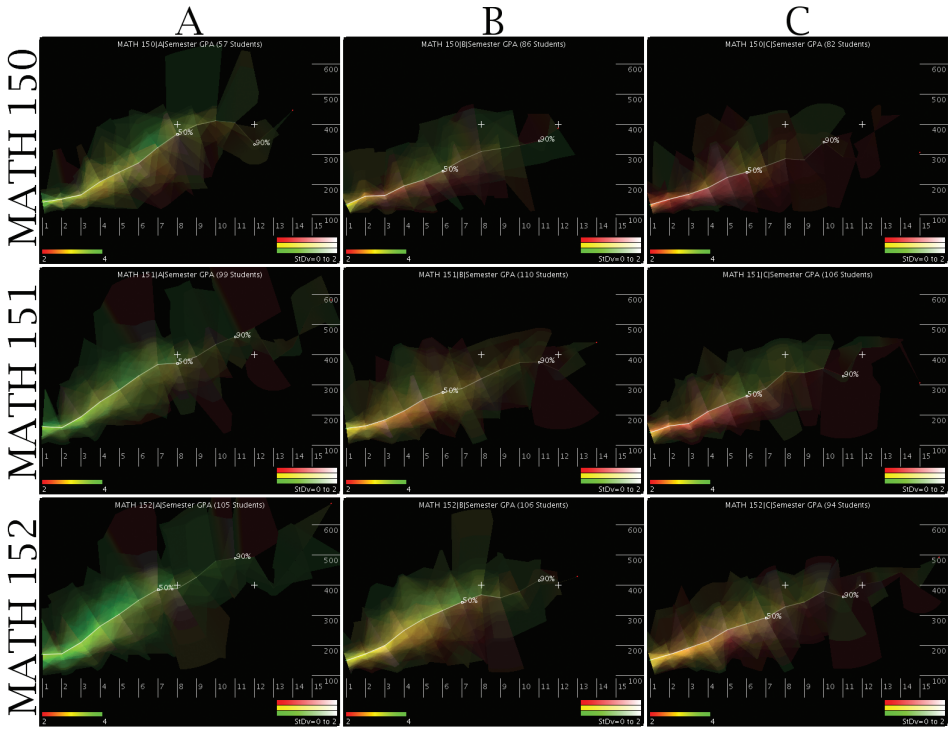

Fig. 9. Core math for computer science students. These initial courses had the most effect on the overall progression of computer science students.

used for a quick preview while more in-depth analysis would require the larger versions. It should be noted that to normalize the contrast across the small multiples shown in this paper, all of the tiles have to be pre-rendered to calculate the means and standard deviations.

5.2 Identifying Plateaus

During discussions with academic subject matter experts, the flat-line trend late in a student's history was identified as particularly interesting. To identify other course-grade pairs that exhibited the same characteristic, the student course history was analyzed using clustering to find commonalities in course-grade trajectories. In order to use clustering, a distance metric had to be defined to characterize the feature of interest—in this case, an initial upward slope followed by either a plateau or decrease in performance toward the end of the student trend line. A distance metric was created to compare the students' trajectory (i.e., slope) irrespective of the coordinate spatial position. To accomplish this goal, each spatial trajectory was broken into three line segments. For each segment, the normal vector was calculated. To compare two trajectories, the root mean square of the angle difference between the two comparison trajectories was calculated.

The items for the clustering algorithm were derived from each possible letter grade and course combination. Each combination was aggregated and the average spatial trajectory was calculated for that specific pair. For example, all students who received an A in CMSC 201 were averaged into the same trajectory and the "CMSC 201 A" trajectory was then compared against all other course grade combinations. Once the course grade combinations were enumerated, hierarchical clustering was used to produce clusters.

Figure 12 shows the main clusters resulting from the clustering algorithm. In the figure, six separate clusters are shown. Within each tile, the course-grade trajectories (colored by grade) and the overall trajectory (white) are shown. In addition to the trajectories, the individual course-grade combinations that create the trajectory are shown in the bottom of each tile. All of the tiles show leveling off or dips late in the students' careers. Cluster 24 is of particular interest since all of the grades show As but a sharp dip occurs at semester 13. A careful examination of the underlying data shows students taking either PHYS 122 or lower numbered math courses—possibly to satisfy degree requirements. Across many of the course-grade trajectories, this was a common theme late in a students career—i.e., taking early requirements. Cluster 21 is also significantly different. Most of the courses represent early general science or engineering topics. As a result of this analysis, educators can revise the curriculum to address identified deficiencies—for instance, by moving core concepts (e.g., math) to precede courses that require that skill set or by providing refreshers.

5.3 Comparing Student Groups

While the small multiples approach reveals differences when the trajectories vary significantly, understanding subtle differences between the student groups is difficult. By modifying the approach to show the differences between two compositions, these variations were able to be more easily detected. To modify the approach, student subsets (e.g., gender, gateway courses) are compared against the entire student population. The composition process remains mostly the same—first, the entire student population is rendered and then the subset is rendered using the same frame of reference (i.e., same average trajectory and parametric mapping) separately. This process results in two separate compositions that can be compared pixel-by-pixel.

To render the difference composition, a Brewer colorscale [6] based on a divergent color scheme of seven values is used. The lower three values show negative differences while the upper three show positive ones. The middle value is used to denote no change. The composition itself still implements the density functionality to provide "dimming" in areas with fewer students.

The results are revealing especially when subtle differences exist. Figure 13 shows both results—the left-hand tiles show the male and female trajectories for the computer science data set using just the composition technique. While the results look very similar, the difference images on the right-hand side reveal subtle variations in the semester grades. While the men had very little differences from the overall student set, the women had slightly higher grades on both sides of the average trajectory (i.e., the blue tones). To draw out the minute differences, a graduated, discrete binning range is used. To show the difference, the population trajectory is depicted in white while the subset trajectory is shown in yellow. For the gender dataset, these are al-

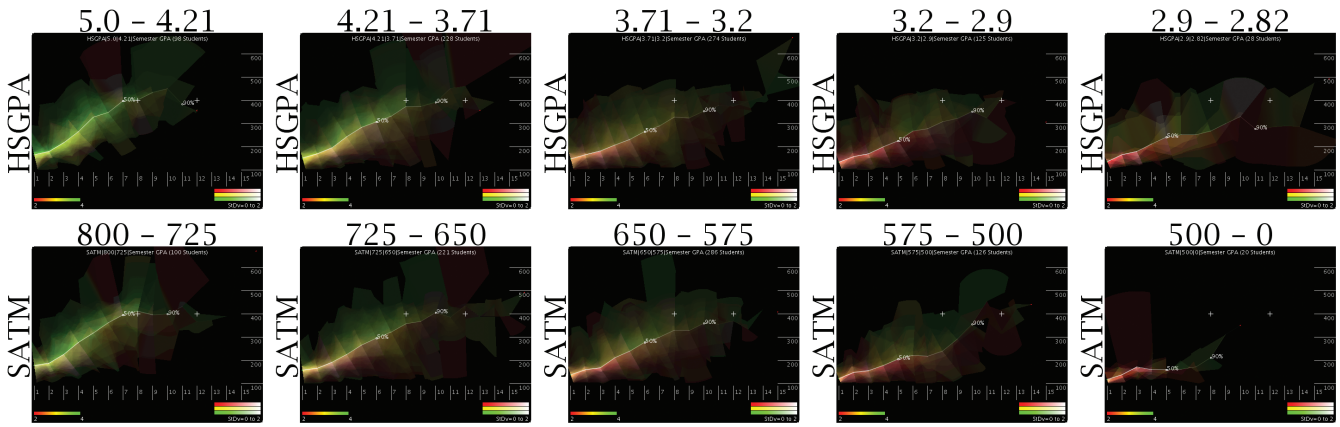

Fig. 10. High school GPA versus SAT Math for computer science students. While the overall tendencies are as expected, high school performance is not significantly more important than SAT testing.

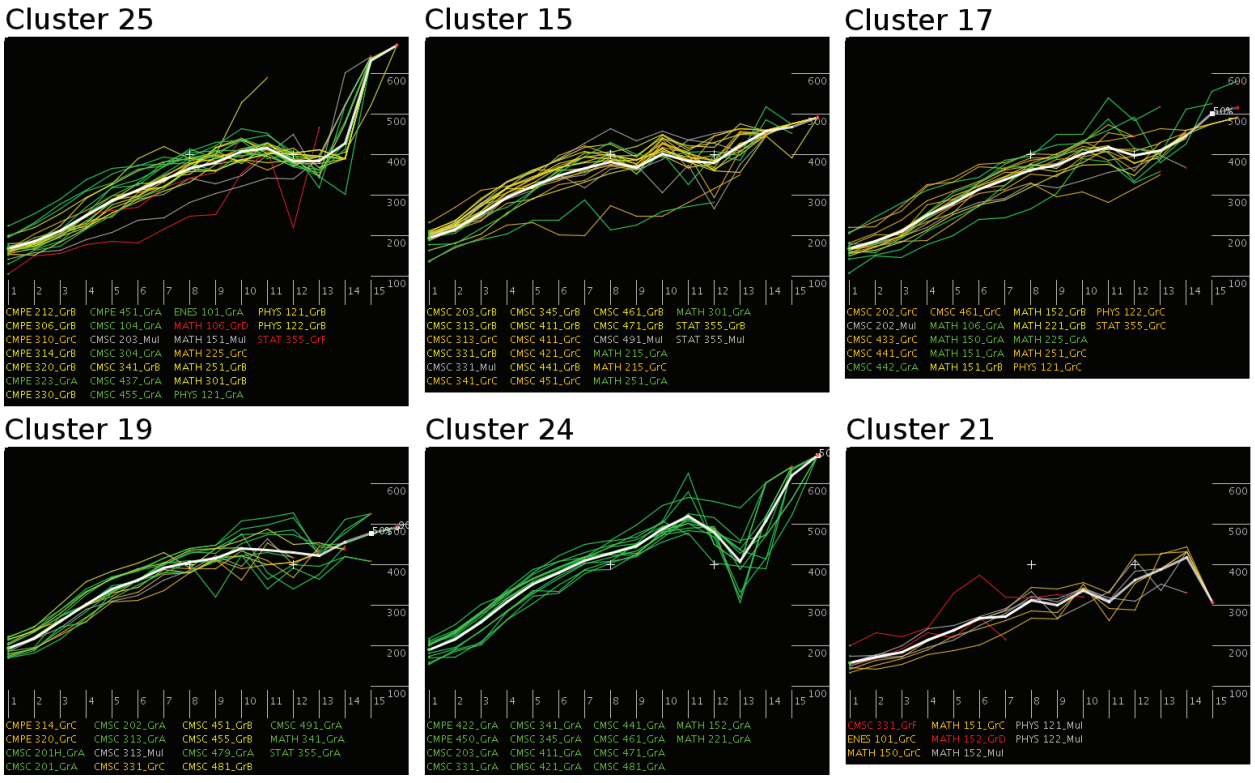

Fig. 12. Cluster results for average course-grade trajectory bins. Note the sudden dips at the end of the most common trajectory shapes—this occured due to students satisfying low prerequisites for graduation.

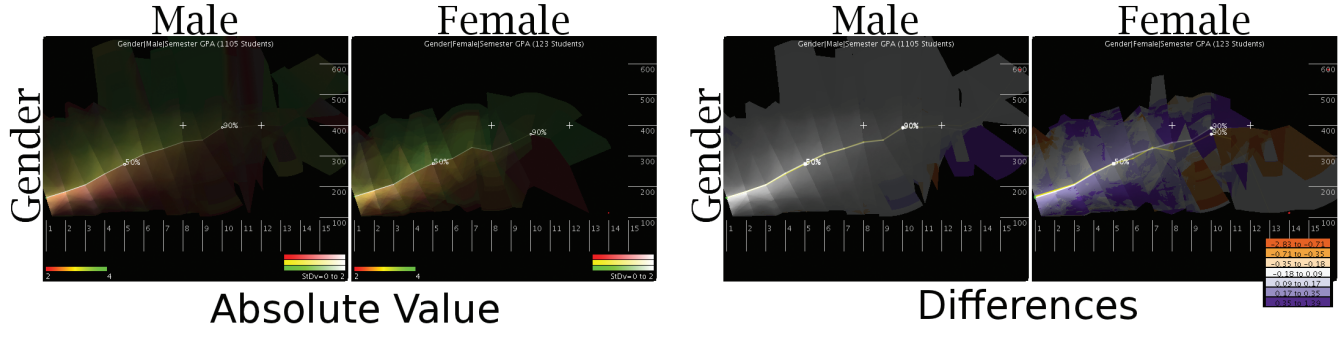

Fig. 13. Computer science gender differences. The left-hand renderings show the results of the semester grades while the right-hand tiles show the difference images. Note that computer science women had higher grades (i.e., blue tones) even though they had slightly lower progressions.

most indistinguishable from the overall population. However, by comparing the overall grades using the differencing approach, the slight improvement of the female students becomes recognizable.

6 Domain Expert Input and Experiences

We solicited input from several educational domain experts during the design and realization phases, including two Undergraduate Program Directors, one Accreditation Coordinator, and one Scholarship Program Director. Their input is reflected in several aspects of the current system, including the capability to show disaggregated groups, the overlay of one group on another, the orientation of the images, and features to support drill-down to specific clusters and trajectories.

These domain experts have also engaged in extended interaction sessions with the completed prototype. While multiple known trends were confirmed during these sessions, unexpected relationships were also discovered in the data. For instance, the importance of a student really doing well in the first computer science gateway course over the second was an unexpected find in the data. This could indicate either that the concepts in the first course were much more critical to success than the second course. Alternatively, the first course could be sufficiently easy that only the highest performers achieved future success. This insight into the critical importance of the first course could be used to identify interventions to help students who don't start off as strongly as they should.

In addition to determining which courses most prominently influenced a student's career, the demographic analysis also revealed additional insights to educators—most notably, the high school GPA comparison with SAT Math. The results were contrary to traditional beliefs that high school GPA is more critical to success than standardized test scores. By identifying these anomalies, educators can delve into the underlying details to understand the nature and cause of the trend. This insight could be used to fine-tune admission standards or to identify students who would benefit from early intervention.

The primary deficiency with the existing approach was the inability for the domain experts to understand the root cause of discovered trends. As noted above, the compositions presented confirmation of known trends and interesting discoveries of unknown aspects in the course histories. For the discoveries, the experts need additional capabilities to explore the dataset in an interactive environment. This need will be addressed in the next section.

7 Limitations, Applicability to other Domains, and Future Work

Multiple issues and limitations became apparent as the results of the composition process were analyzed. The primary challenge was identifying how to group the students to reveal trends of interest. While a small multiples approach was used to organize the visualizations according to course-grade combinations, this process required manual analysis to identify differences and important anomalies. Furthermore, the clustering of students by their course history did not reveal as many insights as hoped. A possible solution would be to cluster the small multiples results. This would group the first stage aggregate trajectory results (i.e., course-history combinations) into partitions to better identify similar average trajectories and anomalies. In effect, this would decrease the manual review and increase the probability that important features were revealed.

Transfer students also presented a challenge. While the start of their trajectories began at the first semester index, calculating their initial offset might have yielded more coherent compositions. For example, because all student trajectories began at the first semester (spatially), transfer students tended to pull the average trajectory higher in the beginning. If, however, a method could be developed and implemented to place them in their true offset, then the composition would not have been skewed by the transfer-student bias.

The last challenge was the visual banding caused by the enforced temporal rigidity of the dataset. This was the result of students taking widely varying course numbers towards the end of their academic careers. These artifacts were also related to the reduced sample size towards the end of the student career—e.g., after the fiftieth percentile marking, the average trajectory was much more likely to fluctuate, due to the small number of students contributing to the average calculation.

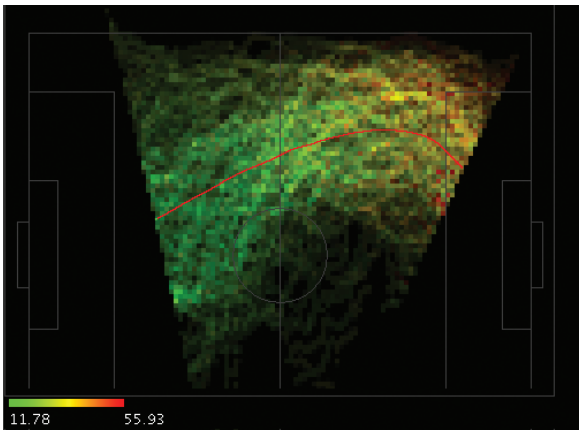

Fig. 14. Multi-agent analysis using composition technique. The figure shows the composite of hundreds of RoboCup soccer plays that have been clustered into similar sets.

In addition to these issues, areas of future work include overcoming the overrepresentation issue and providing interactivity to explore the dataset once a discovery is made. While the level set approach resolved areas of contention, overrepresentation occurred on the outer edge of curves in the average trajectory. This occurred due to the repeated parametric value in the level set calculation. A post-processing step could be developed to smoothly interpolate these values across the outside bends resulting in a smoother composition that revealed additional details and more diverse data.

As noted by the domain experts, interactivity is absolutely essential to explore the underlying causes of the aggregate trends. For our efforts, we created additional GUI interfaces to explore the dataset. For example, in addition to creating the composition based on course grade combinations, we also displayed histograms colored by grade for all of the students within a combination. The histograms did sometimes enable us to find the underlying causes. To achieve near-realtime interactive capability, a small composition size can be used for reasonable numbers of trajectories (in the hundreds). This could then be combined with linked views or brushing techniques [15] to enable faster conclusions on identified trends. For larger composition requirements, space-time trade-offs can be implemented to reduce the number of passes required to create the composition.

Lastly, the composition process can also be applied to arbitrary two-dimensional trajectories—specifically trajectory sets that require analysis of the entities creating the trajectories. While the averaging trajectory process requires modification, the technique has been successfully applied to multi-agent simulation data. Figure 14 shows the results from a multi-agent RoboCup tournament (artificial intelligence robotic soccer competitions [14]). In the figure, the ball trajectories have been clustered into similar shapes and then composited together. The resulting visualization depicts the average distance to the offensive soccer team. In this example, the offensive team is moving the ball to the outside of the field to defeat the defense's tactics—note the gradual change to yellow as the ball progresses down the field. Additionally, the density shows the overall shape of the trajectory and where trajectories varied significantly from the average trajectory.

8 Conclusion

This paper provides an approach for analyzing student course data using logical groupings and composite visualizations. In addition to developing an intuitive spatial representation, an aggregate composition scheme for trajectories is also proposed. Based on feedback from educators, clustering techniques are developed to identify similar histories. Furthermore, a difference composition technique is also implemented which shows minute differences between student groups.

REFERENCES

[1] W. Aigner, S. Miksch, H. Schumann, and C. Tominski. *Visualization of Time-Oriented Data*. Human-Computer Interaction Series. Springer, 2011.

[2] C. Brewer. Color use guidelines for data representation. In *Proceedings of the Section on Statistical Graphics*, pages 55–60, 1999.

[3] E. Chlan and P. Rheingans. Multivariate glyphs for multi-object clusters. *Information Visualization, 2005. INFOVIS 2005. IEEE Symposium on*, pages 141–148, Oct. 2005.

[4] S. H. Edwards, M. A. Pérez-Quiñones, M. Phillips, and J. RajKumar. Graphing performance on programming assignments to improve student understanding. page 6, San Juan, Puerto Rico, 07/2006 2006.

[5] H. Hagh-Shenas, S. Kim, V. Interrante, and C. Healey. Weaving versus blending: A quantitative assessment of the information carrying capacities of two alternative methods for conveying multivariate data with color. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1270–1277, 2007.

[6] M. A. Harrower and C. A. Brewer. **ColorBrewer.org**: An online tool for selecting color schemes for maps. *The Cartographic Journal*, 40:27–37, 2003.

[7] J. Heinrich and D. Weiskopf. Continuous Parallel Coordinates. *IEEE Transactions on Visualization and Computer Graphics (Proceedings Visualization / Information Visualization 2009)*, 15(6), 2009.

[8] R. Mazza and V. Dimitrova. Visualising student tracking data to support instructors in web-based distance education. In *Proceedings of the 13th International World Wide Web Conference on Alternate Track Papers & Posters*, WWW Alt. '04, pages 154–161, New York, NY, USA, 2004. ACM.

[9] P. Muigg, M. Hadwiger, H. Doleisch, and M. E. Gröller. Visual coherence for large-scale line-plot visualizations, June 2011.

[10] U. Rueda, M. Larraaga, M. Kerejeta, J. Elorriaga, and A. Arruarte. Visualizing student data in a real teaching context by means of concept maps. In *Proceedings of the 5th International Conference on Knowledge Management (I-Know 2005)*, pages 561–569, Graz, Austria, 2005.

[11] J. A. Sethian. *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science*. Cambridge University Press, 2 edition, June 1999.

[12] S. Silva, J. Madeira, and B. Santos. There is more to color scales than meets the eye: A review on the use of color in visualization. In *Information Visualization, 2007. IV '07. 11th International Conference*, pages 943 –950, july 2007.

[13] E. R. Tufte. *Envisioning information*. Graphic Press, 1990.

[14] U. Visser and H.-D. Burkhard. Robocup: 10 years of achievements and future challenges. *AI Magazine*, 28(2):115–132, 2007.

[15] R. Voigt. An extended scatterplot matrix and case studies in information visualization. published as diplomarbeit. Master's thesis, Oct. 2002.

[16] D. Wortman and P. Rheingans. Visualizing trends in student performance across computer science courses. *Proceedings of the 38th Annual ACM Technical Symposium on Computer Science Education*, 39:430–434, March 2007.