EG UK Computer Graphics & Visual Computing (2018)
G. Tam and F. Vidal (Editors)

GPU-Assisted Scatterplots for Millions of Call Events

D. Rees[1], R. C. Roberts[1], R. S. Laramee[1], P. Brookes[2], T. D'Cruze[2], and G. A. Smith[2]

[1]Swansea University, UK
[2]QPC Ltd, UK

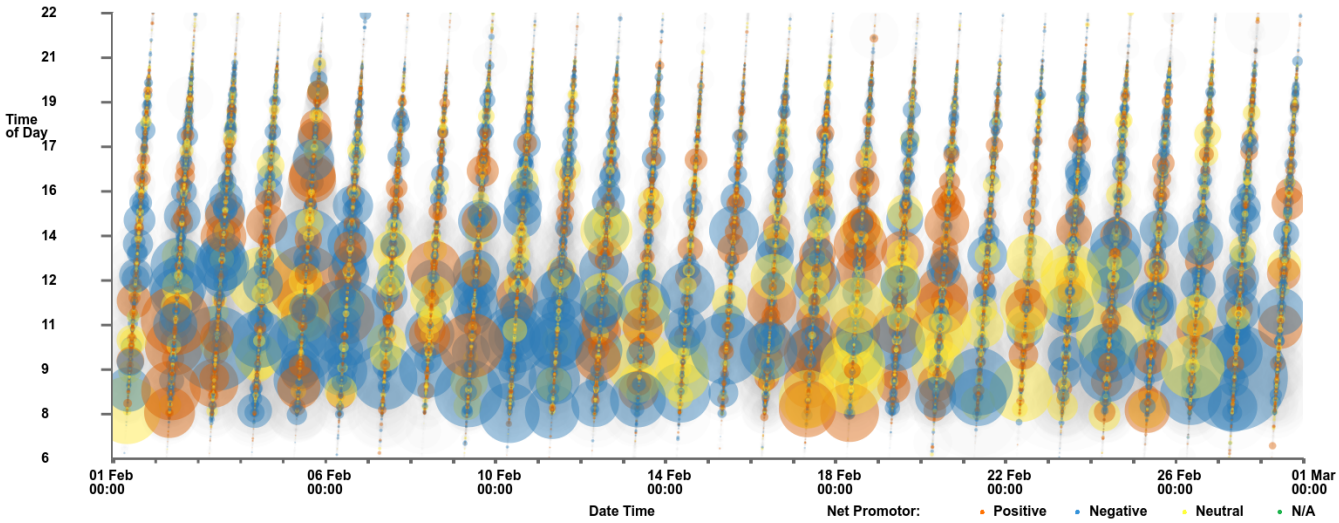

Figure 1: *Scatterplot with call time on the y-axis and call date and time on the x-axis. Point size is mapped to call duration and color to customer feedback score. Data points that represent calls that do not have a feedback score are filtered out. A clear trend is observable: the majority of the longer calls with the largest points are at times before 1400, with shorter calls in the afternoons and evenings.*

Abstract

With four percent of the working population employed in call centers in both the United States and the UK, the contact center industry represents a sizable proportion of modern industrial landscapes. As with most modern industries, data collection is de rigueur, producing gigabytes of call records that require analysis. The scatterplot is a well established and understood form of data visualization dating back to the 17^{th} century. In this paper we present an application for visualizing large call centre data sets using hardware-accelerated scatterplots. The application utilizes a commodity graphics card to enable visualization of a month's worth of data, enabling fast filtering of multiple attributes. Filtering is implemented using the Open Computing Language (OpenCL), providing significant performance improvement over traditional methods. We demonstrate the value of our application for exploration and analysis of millions of call events from a real-world industry partner. Domain expert feedback from our industrial partners is reported.

CCS Concepts

•***Human-centered computing*** → ***Information visualization;*** *Visualization toolkits;*

1. Introduction and Motivation

Contact centers are important contributors to the global economy with 3.6 million contact center agent positions working in 40,750 contact centers within the United States, representing 4% of the working population [Con18b]. The UK meanwhile has an additional 6,200 contact centers with 770,000 agent positions, also representing 4% of UK's working population [Con18a]. This is set to increase with a recent survey revealing that 67.8% of contact

centre operators forecast an uplift in the number of overall interactions [Dim16]. Contact centers are an important part of many industries as a method of interfacing with customers.

Four out of five organizations recognize customer experience as a key differentiator between them and their competitors and over three quarters of companies rank customer experience as the most strategic performance measure [Dim16]. Better customer experience also has financial benefits with 77% of organizations able to evidence cost savings from its improvement [Dim16].

Traditionally call center metrics have centered around service times, queue wait times, abandonment rate and other similar metrics [AAM07]. However customer experience is a multifaceted metric with many influences that span multiple interactions between the organization and the customer. Customer relationship management systems are used to capture and store information related to customer interaction with a company. The use of these systems has also been shown to decrease overall call volume [MG06]. To further improve call center performance, it is important to continue to collect and analyze call records. Data collection is often performed by call center operations systems. However, with multiple attributes of each call recorded and a high call volume, the amount of data becomes difficult to analyze.

Data visualization and visual analysis provide an effective means of analyzing data and to discover insight into behavior. In this paper, we present techniques and an application for visualizing a large multi-call center data set. We demonstrate our application with a data set comprising of almost 5,000,000 calls collected over a month, with each call described by over 70 attributes including over 32 million events. Our application design is based on Shneiderman's visual information-seeking mantra of overview first, zooming and filtering, and details on demand [Shn96]. We present visual designs that enable the linking of calls associated with individual customers to track each customer journey. We also demonstrate the use of GPU computation for enabling fast filtering and rendering of large data sets filtered by multiple attributes. Our contributions are:

- A novel interactive scatterplot application that visualizes 5,000,000 calls
- The ability to track customers over multiple calls
- Advanced interactive and hardware accelerated filtering of call and customer parameters
- The reaction and feedback from partner domain experts in the call center industry

The remaining sections of this paper are set out as follows: Section 2 details work related to this topic, including call center operations management, hardware acceleration, and scatterplot applications. Section 3 details the data set while Section 4 outlines the features and implementation of the application. Domain expert feedback is presented in the Section 5. A conclusion is drawn in Section 6.

2. Related Work

A recent survey of surveys in information visualization by McNabb and Laramee [ML17] offers a helpful starting point for related literature. Scatterplots are a long established method of data visualization that date back to the 17^{th} century, as described by Friendly and Denis [FD05]. However, the visual design has limitations when it comes to plotting large amounts of data. Some modifications, by methods such as subsampling, binning and clustering, have been proposed to overcome the limitations due to large numbers of points. Ellis and Dix survey clutter reduction methods for information visualization [ED07]. Methods explored include clustering, sampling, filtering, use of opacity, differing point sizes, spatial distortions and temporal solutions. Sarikaya and Gleicher also survey scatterplot techniques and identify which design options are best suited to different scatterplot tasks [SG18]. The paper first outlines analysis task performed with scatterplots before examining different data characteristics. A taxonomy of scatterplot designs is presented with reference to suitable tasks and data characteristics. A binning technique to reduce clutter is introduced by Carr *et al.* [CLNL87]. They demonstrate the use of hexagonal bins with the size and color of each bin proportional to the number of points. Keim *et al.* propose a space distortion technique to minimize overlap of data points [KHD*10]. The user is able to control the level of overlap and distortion to view trends in the data. Deng *et al.* introduce a technique for visualizing overlapping data by stacking elements in a third dimension [DWA10]. To overcome overplotting, Chen *et al.* use a sampling method to form a cloud that represents multi-class point distributions [CCM*14]. Mayorga and Gleicher use a kernel-density estimation of multi-class data to visualize dense regions as contour bounded areas [MG13]. The technique presented also supports the use of GPU computation to enable interaction with large data sets comprising of up to three million data points.

2.1. Information Visualization and Hardware Acceleration

A GPU implementation of an adjacency matrix graph is presented by Elmqvist *et al.* [EDG*08] where 500,000 French Wikipedia pages are represented with 6,000,000 links. McDonnel and Elmqvist [ME09] present a refinement to the traditional information visualization pipeline, to incorporate the use of GPU shaders, enabling the use of parallel computing and interactive plotting of large data sets. The technique is shown to be applicable to many visual designs including treemaps and scatterplots. They postulate that this is due to a gap between the abstract data types requiring visualization and the GPU shader languages that would be used. To remedy this they present a visual programming environment that generates the required shader code. Mwalongo *et al.* review web-based visualization applications which utilize GPU-based technologies such as WebGL to render large data sets [MKRE16]. Technologies are categorized according to their application domain with categories covering the scientific visualization, geovisualization, and information visualization fields. The survey features three publications which utilize hardware acceleration to process and render scatterplots [LJH13, AW14, SGC*15]. These publications however only feature small data sets or use an aggregation pre-processing step to reduce the number of data points, whereas we demonstrate filtering and rendering of almost five million data points. These papers rely on web-based technologies such as JavaScript and WebGL, while we concentrate on local GPU computation.

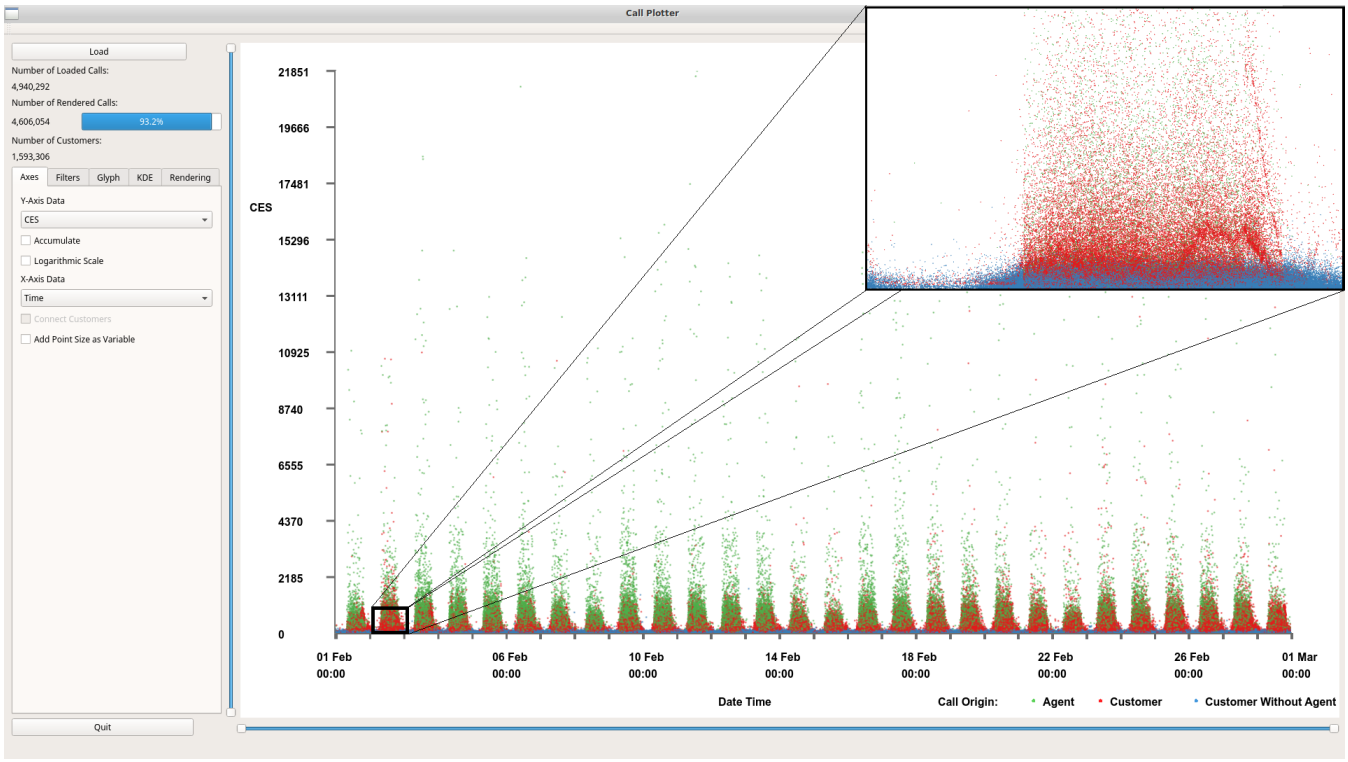

Figure 2: *Overview of application interface with one month of call data loaded. The default view shows the CES on the y-axis and the time of the call on the x-axis, to this we have added a zoomed in image for one day. Calls are colored by their origin, green indicates an agent initiated the call, red that the customer initiated the call, and blue indicates a customer initiated call with no agent interaction. Immediately we can see the periodic pattern of calls spanning a month where peak times are mid-day every day. Within the zoomed framed, zoomed to approximately one day, an interesting pattern in the data is seen.*

2.2. Call Center Analysis Literature

Call center operations are a complex subject with many intricacies. For a comprehensive overview we recommend that readers consult *"Call Center Operation: Design, Operation, and maintenance"* by Sharp [Sha03]. The demands on a call center can be difficult to predict even with research studying incoming call rate [JK01, WBS07, BGM*05]. This creates a difficult challenge for call center managers who have to balance costs and the staffing levels required to cope with the call demand. Failure to achieve a correct balance can lead to either high staffing costs or dissatisfied customers with long waiting times trying to contact the call center. Due to the complex nature of call center management, there exists a large body of research into the challenges that they face. Askin *et al.* provide a comprehensive survey of the research up to 2007 [AAM07]. The paper is organized into different areas of call center management surveying traditional call centre operations, research into call demand modulation, effect of technological innovation, human resource issues, and the integration between call center operations and marketing.

A statistical analysis of call center data is presented by Brown *et al.* [BGM*05]. Three service process are explored: call arrival, customer patience, and service duration. Shi *et al.* demonstrate improvement of a telephone response system in a veterans hospital [SEP*15]. Roberts *et al.* presents an interactive treemap application for displaying call metrics of calls serviced at a call center over one day [RTL*16]. Roberts *et al.* also use the same data to demonstrate a higher-order brushing technique for parallel coordinate plots [RLS*ng]. Their data set is limited to one day only while this work can handle one month's worth of data.

3. Call Center Data characteristics

The data has been collated in a database developed by our partner company QPC Ltd. and consists of all calls to one of their client's call centers during February 2015. All calls have been anonymized. In total there are 4,940,292 calls collected from 43 different sites across Egypt, India, Romania, South Africa, and the UK. The data set consists of four separate CSV files, each file consisting of different attributes linked by a common *'Connection Identifier'* to link the individual calls.

Each call has over 70 attributes, some are recorded directly such as the call duration, whilst others are derived, such as the call cost to the operator. Other attributes are used to identify the customer, the agent(s) spoken to and the site where the agent is based. Each call is initially received by an interactive voice response system (IVR). This is an automated menu system that plays a prerecorded message, then directs the call according to the inputs from the caller.

Two important measures of customer satisfaction are supplied as part of the data set: customer effort score (CES) and net promoter score (NPS). CES is a derived metric that tries to establish how

much effort a customer has applied in each call, with a lower score indicating that the call is easier for the customer. Some factors that contribute to the CES are the call duration, wait duration, the number of agents spoken to, and the number of transfers. The NPS is only supplied for a small percentage of the calls (3.7%), involving a post-call survey sent to the customer, and completed by the customer. The NPS value is a score out of ten of how satisfied the customer was with the call, with 10 indicating very satisfied and 0 extremely dissatisfied.

4. Hardware Accelerated Scatterplots

The software is written in C++ using the Qt framework (version 5.9) [The95] and OpenGL (version 4.5). Development was performed on a Ubuntu 18.04 system with an Intel i7-6700k processor, 16GB of RAM and an Nvidia GTX1070 graphics card. The software was also tested on a Windows system with an Intel i7-6700HQ processor with 8GB of RAM and an Nvidia GTX1060 6GB mobile graphics card. The software must first import and process the data before the graphics can be constructed. Processing the data predominantly involves connecting the calls from across different files, and linking calls to customers to facilitate look-up. The default view of the application, once data has been pre-processed, can be seen in Figure 2, displaying over 4.6 million calls. The daily periodicity of call volume is immediately conveyed. The main window of the application shows the scatterplot chart, with a side panel for various interaction and filtering options, based on Shneiderman's visual information-seeking mantra [Shn96]. These interaction options include:

- Fully interactive zooming on two independent axes
- User-chosen axis variables
- GPU enhanced filtering of multiple call attributes
- Brushing data points for details on demand

Due to the large volume of data, these interaction options are important to enable exploration of the data. Filtering is provided for a number of call attributes and is split into two categories, customer filters, for customer-oriented attributes such as accumulated CES, and call filters, for call related filters such as call duration. To garner more information about a particular data point, the user is able to brush the point with the mouse which activates a dialog containing details about the call.

Figure 2, shows an overview of the loaded data set. Notable within the figure is the layered nature of the colors representing the call origin. The calls that do not involve a call center agent (blue - conspicuous in zoomed section), are predominantly at the bottom with the lowest CES, calls initiated by a call center agent (green) generally have a higher CES with the customer initiated calls (red) sandwiched in between. The total number of calls loaded is shown along with the number of calls displayed and a bar displaying the percentage of loaded calls rendered. Also the number of customers represented in the scene is given. Within the scatterplot the call volume distribution can be observed, a peak of calls can be seen at each day with troughs at night times. The majority of the data can be seen in the lower areas of the scatterplot space, with proportionately less calls in the upper two thirds.

4.1. Scatterplots View

The default view depicts the CES of each call along the left y-axis against the time the call was made along the x-axis, as can be seen

Figure 3: *A zoomed scatterplot with supplementary call metric lines. Wait duration is represented on the y-axis against the date and time on the x-axis. The zoom function is used to zoom in to a single day and to a wait duration of below 30 minutes on the y-axis. Calls are colored by their origin. Call metrics lines are also shown. The majority of calls can be observed between 0800 and 2100 indicating the times where the main call centers are open. A gap can be seen between 1700 and 1800 indicating a malfunction with either data recording or call center operations. An increase in the waiting times for customers can be observed between 0700 and 0800.*

in Figure 2. Color is mapped to call origin. Green indicates an agent initiated the call, red the customer initiated the call, and blue indicates a customer initiated call without any agent interaction. An agent interaction might not occur due to the call requirements being served by the IVR or because the customer abandoned the call. The user is able to click on the color key to choose from a selection of other color-maps if required. A drop-down menu is available for each axis, to change the axis variables. Options for the y-axis include CES, call cost, call duration, agent duration, wait duration, IVR duration, hold duration and time of day of the call. These call attributes are also available for the x-axis, along with additional attributes of date and time of the start of the call, date and time of the end of the call, and a normalized call date and time. The normalized time is based on the time since the first call of each customer in the data set.

Interaction: The user is able to smoothly zoom in on particular regions of the scatterplot by either using the mouse wheel or sliders at the edge of the plot area. Each axis is zoomed independently with the mouse wheel zooming in on the x-axis only and the control modifier used in conjunction with the mouse wheel to zoom on the y-axis. Users are able to explore the scatterplot by clicking and dragging the zoomed scene. Figure 3 shows a zoomed scene, with zooming on both the x and y axes.

Rendering options: The user also has the option to map the size of the data points to a third call attribute to enable further exploration, as can be seen in Figure 4. Figure 4 also shows calls connected by a polyline. This polyline is another user option and connects multiple calls that are made from the same customer. To establish a customer's satisfaction with the service they receive, it is important to consider all interactions that the customer makes and not treat each call in isolation. To aid the exploration of this, we enable the user to accumulate the CES and cost for each customer.

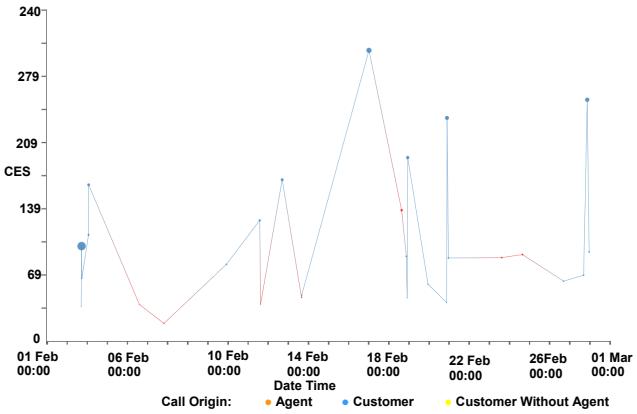

Figure 4: *The CES on the y-axis and the time of the call on the x-axis for all of the calls for a single customer. Point size is proportional to call duration. Calls are connected with a line to indicate all calls are made by the same customer. Calls are colored by their origin.*

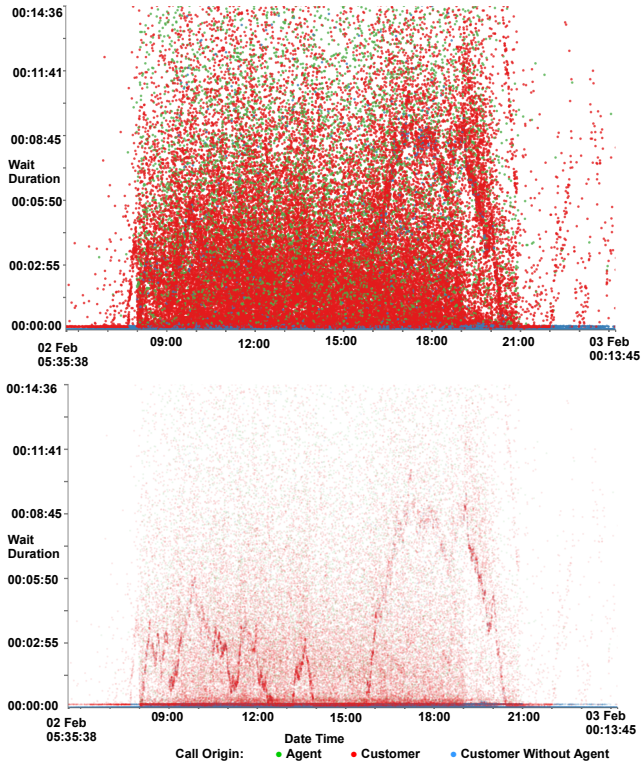

Figure 5: *A comparison of plot opacity for the wait time plotted against date and time of calls. By reducing opacity (bottom) different patterns become visible within the data set compared to the patterns seen with high opacity (top).*

This is achieved by ordering all calls from a particular customer chronologically and accumulating the totals for each call. Users also have the option to adjust the size and opacity of the data points for easier exploration. In sparsely populated scatterplots, larger data points are easier to distinguish, whilst in over-plotted data smaller points prevent clutter. In overplotted areas of data, reducing the opacity of the data points enables discovery within dense data regions. This can be seen in Figure 5 where the reduced opacity image shows a pattern in the data that was previously hidden.

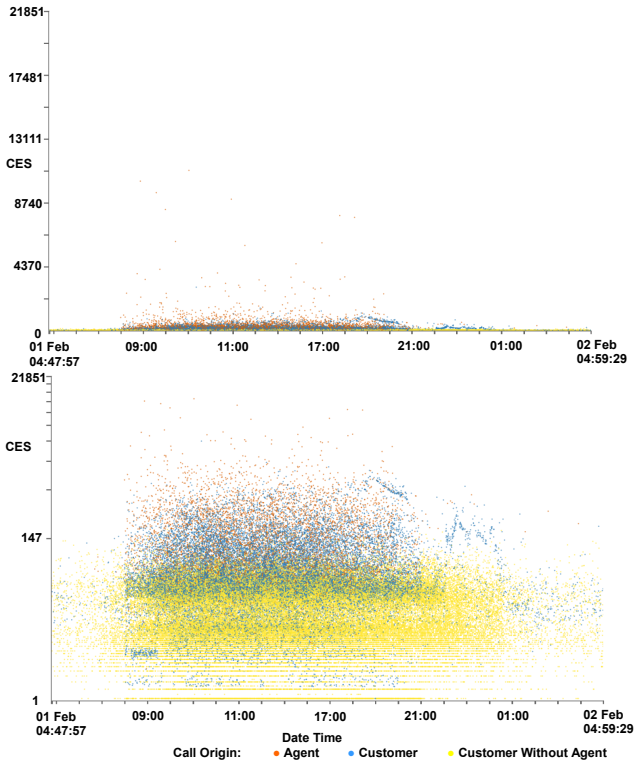

Figure 6: *Two scaterplots showing the same data, one with a standard linear axis scale (top) and the other using a logarithmic y-axis (bottom). CES is on the y-axis and the time of the call on the x-axis. The x-axis is focused on a single day and calls are colored by their origin. The three layered trend seen in Figure 2 is more visible here, with customers who don't interact with an agent predominantly with lower CES, agent initiated calls with the highest CES and customer initiated calls in between.*

Users also have the ability to adjust opacity for context calls. Calls that have been filtered out are shown in a faded gray to provide context as described by Card *et al.* [CMS99] For more detail on filtering see Section 4.2. Filtered context call data points are also rendered before focus call data points in a two-pass rendering. The first pass renders only context calls while a second pass renders only focus calls. This enables focus calls to be rendered on top of context calls, as can be seen in Figure 9.

It has been found, with initial exploration of the data, that the majority of data points reside in the lower data ranges of CES and cost variables. To enable better exploration of this data we include an option to implement a logarithmic scale, allowing a focus to be put on this data. This is shown in Figure 6.

Caller line plots: Call center metrics are provided to help identify features discovered in the data set as can be seen in Figure 3. Metrics provided are call arrival rate, call abandonment rate, and average waiting time. Call arrival rate is calculated by summing the number of calls every minute, this is then smoothed using a non-parametric regression function on a day-by-day basis, as outlined by Brown [Bro03]. Call abandonment is calculated using the same

technique. Average wait time is calculated using a tricube function with bandwidths automatically chosen using cross-validation on a day-by-day basis, as described by Brown *et al.* [BGM*05]. To supplement this, a typical day line for the wait time, call arrival rate and abandonment can also be shown. The typical day line is constructed by calculating the average day from the month's worth of data. Because arrival rate is significantly different over the weekend compared to the weekdays, typical arrival rate has been separated into weekday values, Saturday values and Sunday values. The typical day metrics can be used as a benchmark and compared to particular days to establish if they are above or below average. This feature allows the observation that Mondays are typically busier than other weekdays and that Thursdays are generally quieter. This can be seen in the supplementary video [10018]. The metric lines can also be used to find benchmarks for comparisons across different data sets from different companies.

GPU implementation We utilize OpenGL to provide the graphical element of the software. Encoding data to axis co-ordinates is pre-computed after the data is loaded. This data is loaded onto the GPU memory buffer and rendered with the use of OpenGL shaders [KSS16]. Using these techniques and a commodity graphics card, we achieve interactive frame rates with almost 5 million data points. The OpengGL fragment shader code is provided below for easy reproducibility.

```
layout(location = 0) in float xVert;
layout(location = 1) in float yVert;
layout(location = 2) in float colorAttr;
layout(location = 3) in float filterAttr;
uniform mat4 matrix;
uniform mat4 cMatrix;
uniform float opacityF;
uniform float conOpac;
uniform float pointSz;
uniform float passNo;
out vec4 color;

void main()
{
    float grey = 0.7;
    vec4 colorAt = vec4(0.0, 0.0, 0.0, 0.8);
    gl_PointSize = pointSz;
    gl_Position =matrix*vec4(xVert, yVert,0.0,1.0);
    //apply color
    if (colorAttr == 2.0) color = cMatrix[1];
    else if (colorAttr == 1.0)
        color = cMatrix[0];
    else if (colorAttr == 0.0)
        color = cMatrix[2];
    else
        color = cMatrix[3];
    //two pass filter
    if (filterAttr == 2.0 && passNo == 0.0)
        color = vec4(grey, grey, grey, conOpac);
    else if (filterAttr == 0.0
            || passNo == 0.0
            || filterAttr == 2.0)
        color = vec4(1.0, 1.0, 1.0, 0.0);
    else color *= vec4(1.0, 1.0, 1.0, opacityF);
}
```

4.2. GPU Enhanced Filtering

To facilitate user-driven exploration of the call data, we have implemented filters for multiple call attributes. Some filtering can be achieved visually using the zoom function, however this is limited in functionality. Two groups of filters are used, customer-based filters and call-based filters. Customer-based filters enable filtering of groups of calls belonging to particular customers using variables over all loaded calls. Available customer filters are:

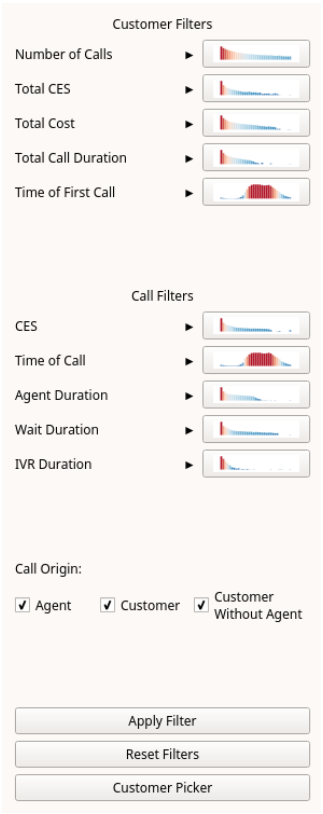

Figure 7: *Filter interface including thumbnail previews with call attribute histograms on buttons. Buttons are highlighted with red when filters are applied.*

- Number of calls made by a given customer
- The total CES accumulated for all calls of a customer
- The total cost of all calls for a customer
- The total call duration accumulated over all calls of a customer
- The time of the first call for a customer

Call filters are used for filtering individual calls. Attributes available for call filters are:

- CES of a call
- Time of day of a call
- The amount of time spent with an agent
- The wait duration of a call
- The IVR duration of a call

An additional filter is available to filter each of the different origins of the calls. Figure 7 shows the user interface to facilitate filtering, with filters split into customer-based and call-based. The distribution of calls can be seen on the thumbnail previews of histograms to aid filtering decisions, with filters that have already been applied highlighted in red. Clicking a filtering button enables the filtering dialog for that attribute (Figure 8 shows the filter dialog for wait duration). The filter dialog shows two histogram plots of the attribute, the topmost shows the total distribution whilst the lower shows the

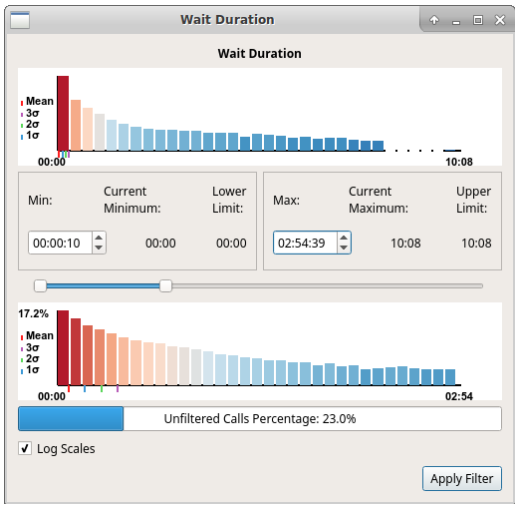

Figure 8: *Filter dialog for wait duration. Two distributions are shown, the top shows the total call distribution and the bottom shows the distribution resulting from the applied filters*

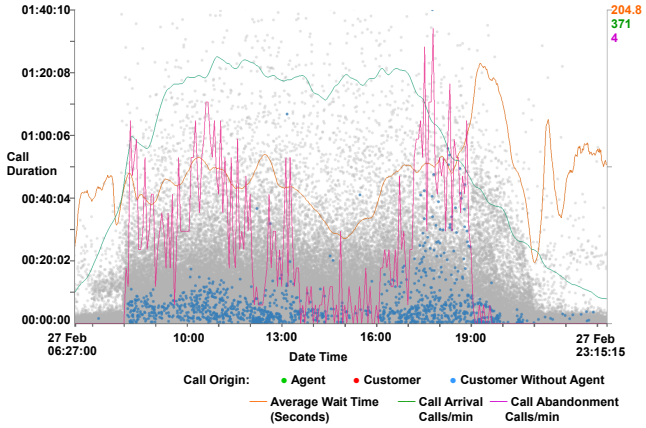

Figure 9: *A filtered scatterplot zoomed to a single day and call duration of less than 1hr 40min including supplementary call metric lines. Filters are used to exclude calls with agent interaction and with a waiting time of less than 10 seconds. Calls excluded from the filter are rendered gray in the background to provide context. Correlation can be seen with the call duration and the call abandoned rate (red line).*

distribution with user-adjustable lower and upper range limits set in the controls applied. A selection box at the bottom of the dialog enables a logarithmic function to be applied to the distribution to better spread the data. Filter limits can be set using three control mechanisms, an input box for the lower limit, an input box for the upper limit, and a range slider enabling adjusting of both lower and upper limits. Controls are connected, with changes in one control reflected in the other controls. Indications of the maximum and minimum filtering values as well as the current applied filter values are also provided. A bar is given at the bottom of call filter dialogs to indicate the percentage of total calls that are displayed as a result of applying the filter.

Filters can be applied individually by clicking the apply button in the dialog for the appropriate filter, or all open filters can be applied by clicking the apply button in the main interface. A "reset filters" option is available to set all filters to their maximum and minimum values and a customer picker is available to choose an individual customer for investigation. Figure 9 shows an example of the visualization with filters applied, along with call metrics. A correlation can be seen with the number of abandoned calls metric line and the call duration of the remaining calls. McDonnel and Elmqvist describe the use of GPU for filtering and visualizing using OpenGL shaders [ME09], however this filtering method fails with calls being grouped by customers and requires image processing to ascertain filtering result metrics. In order to remedy this issue, we utilize the parallel processing benefits of a GPU and the Open Computing Language (OpenCL version 2.0) [Mun09] to quickly filter the number of calls and to return the filtering metrics.

To filter the calls without hardware acceleration would require iterating through each call for each customer, testing if each variable is within filtering limits. With millions of calls this method can take minutes to complete. However with the use of parallelism, on the GPU, each call can be tested concurrently. For further instruction on the use of OpenCL We recommend the books by Munshi *et al.* and Scarpino [MGMG11, Sca11]. OpenCL functions, known as kernels, are performed on each instance of the data, in this case calls, returning an output. This can be quickly processed to return the number of calls and customers filtered. Our abridged kernel code can be seen below for reproducibility:

```
int row = get_global_id (0);
if (colIn[row] > 0.0f)
{
   if (inputD[row] > maxD || inputD[row] < minD ||
       inputE[row] > maxE || inputE[row] < minE ||
       inputF[row] > maxF || inputF[row] < minF ||
       inputG[row] > maxG || inputG[row] < minG ||
       inputH[row] > maxH || inputH[row] < minH ||
       inputI[row] > maxI || inputI[row] < minI ||
       inputJ[row] > maxJ || inputJ[row] < minJ ||
       inputK[row] > maxK || inputK[row] < minK ||
       inputL[row] > maxL || inputL[row] < minL ||
       inputM[row] > maxM || inputM[row] < minM ||
       binaryV == 0 ||
       (binaryV%2==0 && binaryL[row]==2) ||
       (binaryV/4<1 && binaryL[row]==0) ||
       ((binaryV==1 ||binaryV==4 ||
       binaryV==5) && binaryL[row]==1))
   {
      sizeOut[row] = 1; //cust filtered out
      colOut[row] = 2.0;
   }
   else
   {
     sizeOut[row] = 0; //cust stayed in
     colOut[row] = 1.0;
   }
}
else
  sizeOut[row] = 2;
outputA[row] = inputC[row];
```

The kernel code tests if each call variable is between the maximum and minimum ranges specified in the filters and outputs the filtered status. Each call is processed with this code, returning a vector of the filtered status of each call. This vector can then be passed to the OpenGL rendering shader so that the data point for the call can be rendered as focus or context. The vector can easily be processed to quickly calculate filtering statistics.

4.3. Brushing for Details

Once particular data points of interest have been identified by the user, they are able to brush the desired region on the scatterplot to

bring up a dialog featuring all attributes of the brushed calls. This fulfills the final part of Shneiderman's visual information-seeking mantra, [Shn96], details on demand. Figure 10 shows an example of the brush dialog. Users are able to copy selected data attributes from the brush dialog for use elsewhere. This copy feature was requested by our domain experts to enable further explorations using different applications.

Video Demonstration: Please visit `https://vimeo.com/270333276` to view a demonstration of the application and its features.

5. Domain Expert Feedback

The software was developed in collaboration with our industrial partner QPC Limited, with whom we have been working with since 2014. The development of this application has been driven by discussions with QPC Ltd. and their requirements. Here we present important feedback garnered at multiple guided interviews after presenting the application and its features.

Initially when shown the software with a month of data loaded, the experts were impressed with the application's ability to plot a large number of data points. When asked if they had seen a month's worth of data before, an expert replied: *"Not at this speed, no. We've had to go down the route of pre-aggregating the data to get the speed."* In fact, this is the first time anyone has seen an entire month's worth of data simultaneously, in their entire company's history. Previous commercial products used to explore the data set have been limited in the size of the input data set. After demonstrating the zooming, panning, and data variable choices, the experts saw the value of the application and the exploration potential it provided: *"It'll be interesting to put a new data set in that we haven't looked at before, that we haven't got any knowledge of and to instantly then be able to see something. Without having to do work to get to really."*

The filtering ability of the software in particular was well received, with the thumbnail previews of histograms exalted for their ability to give an initial summary of the different fields and distributions. *"I like the look of that, it looks nice first of all, it's giving you a good summary of the different fields and distributions."* The ability to compound the filters and the briskness of the filters were also praised. *"You've given the ability to filter the contacts in quite a few different ways and to enable you to focus in on particular areas and for the individual contacts you come down to you can look closer, maybe in a different application."* Positive feedback was also received for the metric and the typical day lines: *"Yeah, I think it's nice, it lets you look at some standard call centre metrics."* The average plot lines were particularly noted for their ability to benchmark call center performance. With this feature our industry partner can, for the first time, compare call center performance between their customers in addition to different days. The ability to brush for individual call attributes was also welcomed, allowing identification of specific identified calls.

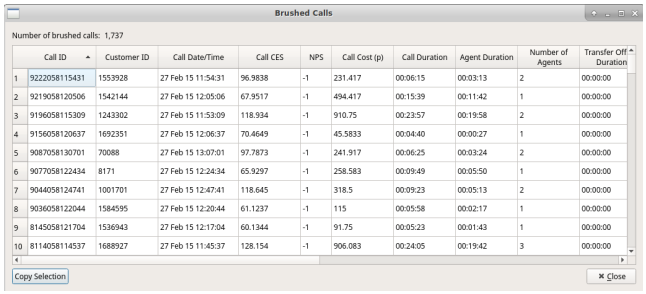

Figure 10: *Brush dialog showing all call attributes for brushed calls in full detail.*

More general feedback was given in regards to the usefulness of the application to QPC Ltd. and their customers: *"I think there are two immediate purposes it serves, one is validation, it'll throw up those outliers we've got... and two, from an insight perspective... we'd probably show this to the customer to demonstrate the insight, to show how flexible the data is."* This was followed up with a statement which we feel encapsulates the aims of the application: *"It makes the application that you've created a stepping stone... because you can look at a large set of data and filter down to a smaller number of calls, this application looks useful for that then potentially you can go and look at some more specific detail with another application or even you just literally go to the database and take those call I.D.'s you've listed out there even just go direct to the database."*

Recommendations for improvements were received from the feedback sessions in particular requested was the ability to include more variables.

6. Conclusions and Future Work

We present an application capable of visualizing millions of calls representing a month's worth of real-world data for the very first time. The application enables fast exploration of a large data set including rapid filtering and brushing for further detail, reflecting Shneiderman's visual information-seeking mantra [Shn96]. Details of fast filtering using OpenCL are presented. Insights into the data set are presented and feedback from our expert industrial partner is also provided.

In future we would like to further explore improvements with the use of general-purpose compute on the GPU. This includes the use of a shared context between OpenCL and OpenGL memory buffers as demonstrated by Alharabi *et al.* [ACL17], and the use of the Vulkan API [The16]. Following feedback from QPC Ltd. we would also like to extend the software to handle more call variables and to utilize dimension reduction techniques to highlight key variables and to find co-relation coefficients. Further testing of the software would also be beneficial with data sets from other vocations, and larger data sets. The ability to display other visual designs is also a desirable feature. Over-plotting is a significant issue when plotting a large number of data points, in future we would like to provide an auto-detection feature for over-plotting that adjusts the opacity accordingly.

7. Acknowledgements

The authors gratefully acknowledge funding from KESS. Knowledge Economy Skills Scholarships (KESS) is a pan-Wales higher level skills initiative led by Bangor University on behalf of the HE sector in Wales. It is part funded by the Welsh Government's European Social Fund (ESF) convergence programme for West Wales and the Valleys. We would also like to thank Liam McNabb and Beryl Rees for their help

References

[10018] 1001 S.: Demonstration video, 2018. URL: https://vimeo.com/272352698. 6

[AAM07] AKSIN Z., ARMONY M., MEHROTRA V.: The modern call center: A multi-disciplinary perspective on operations management research. *Production and operations management 16*, 6 (2007), 665–688. 2, 3

[ACL17] ALHARBI N., CHAVENT M., LARAMEE R. S.: Real-Time Rendering of Molecular Dynamics Simulation Data: A Tutorial. In *Computer Graphics and Visual Computing (CGVC)* (2017), Wan T. R., Vidal F., (Eds.), The Eurographics Association. 8

[AW14] ANDREWS K., WRIGHT B.: FluidDiagrams: Web-Based Information Visualisation using JavaScript and WebGL. In *EuroVis - Short Papers* (2014), Elmqvist N., Hlawitschka M., Kennedy J., (Eds.), The Eurographics Association. 2

[BGM*05] BROWN L., GANS N., MANDELBAUM A., SAKOV A., SHEN H., ZELTYN S., ZHAO L.: Statistical analysis of a telephone call center: A queueing-science perspective. *Journal of the American statistical association 100*, 469 (2005), 36–50. 3, 6

[Bro03] BROWN L. D.: Empirical analysis of call center traffic. In *Call Center Forum* (2003). 5

[CCM*14] CHEN H., CHEN W., MEI H., LIU Z., ZHOU K., CHEN W., GU W., MA K.-L.: Visual abstraction and exploration of multi-class scatterplots. *IEEE Transactions on Visualization and Computer Graphics 20*, 12 (2014), 1683–1692. 2

[CLNL87] CARR D. B., LITTLEFIELD R. J., NICHOLSON W., LITTLEFIELD J.: Scatterplot matrix techniques for large N. *Journal of the American Statistical Association 82*, 398 (1987), 424–436. 2

[CMS99] CARD S. K., MACKINLAY J. D., SHNEIDERMAN B.: *Readings in information visualization: using vision to think*. Morgan Kaufmann, 1999. 5

[Con18a] CONTACTBABEL: *UK Contact Centres: 2018-2022 The State of the Industry & Technology Penetration*. Tech. rep., ContactBabel, 2018. 1

[Con18b] CONTACTBABEL: *US Contact Centres: 2018-2022 The State of the Industry & Technology Penetration*. Tech. rep., ContactBabel, 2018. 1

[Dim16] DIMENSION DATA: *Global contact centre benchmarking report*. Tech. rep., Dimension Data, 2016. 2

[DWA10] DANG T. N., WILKINSON L., ANAND A.: Stacking graphic elements to avoid over-plotting. *IEEE Transactions on Visualization and Computer Graphics 16*, 6 (2010), 1044–1052. 2

[ED07] ELLIS G., DIX A.: A taxonomy of clutter reduction for information visualisation. *IEEE Transactions on Visualization and Computer Graphics 13*, 6 (2007), 1216–1223. 2

[EDG*08] ELMQVIST N., DO T.-N., GOODELL H., HENRY N., FEKETE J.-D.: Zame: Interactive large-scale graph visualization. In *IEEE Pacific Visualization Symposium, 2008. PacificVIS'08.* (2008), IEEE, pp. 215–222. 2

[FD05] FRIENDLY M., DENIS D.: The early origins and development of the scatterplot. *Journal of the History of the Behavioral Sciences 41*, 2 (2005), 103–130. 2

[JK01] JONGBLOED G., KOOLE G.: Managing uncertainty in call centres using poisson mixtures. *Applied Stochastic Models in Business and Industry 17*, 4 (2001), 307–318. 3

[KHD*10] KEIM D. A., HAO M. C., DAYAL U., JANETZKO H., BAK P.: Generalized scatter plots. *Information Visualization 9*, 4 (2010), 301–311. 2

[KSS16] KESSENICH J., SELLERS G., SHREINER D.: *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 4.5 with SPIR-V*. Addison-Wesley Professional, 2016. 6

[LJH13] LIU Z., JIANG B., HEER J.: immens: Real-time visual querying of big data. *Computer Graphics Forum 32*, 3pt4 (2013), 421–430. 2

[ME09] MCDONNEL B., ELMQVIST N.: Towards utilizing GPUs in information visualization: A model and implementation of image-space operations. *IEEE Transactions on Visualization and Computer Graphics 15*, 6 (2009), 1105–1112. 2, 7

[MG06] MEHROTRA V., GROSSMAN T.: *New processes enhance cross-functional collaboration and reduce call center costs*. Tech. rep., Working Paper, Department of Decision Sciences, San Francisco State University, 2006. 2

[MG13] MAYORGA A., GLEICHER M.: Splatterplots: Overcoming overdraw in scatter plots. *IEEE Transactions on Visualization and Computer Graphics 19*, 9 (2013), 1526–1538. 2

[MGMG11] MUNSHI A., GASTER B., MATTSON T. G., GINSBURG D.: *OpenCL programming guide*. Pearson Education, 2011. 7

[MKRE16] MWALONGO F., KRONE M., REINA G., ERTL T.: State-of-the-art report in web-based visualization. *Computer Graphics Forum 35*, 3 (2016), 553–575. 2

[ML17] MCNABB L., LARAMEE R. S.: Survey of surveys (sos)-mapping the landscape of survey papers in information visualization. In *Computer Graphics Forum* (2017), vol. 36, Wiley Online Library, pp. 589–617. 2

[Mun09] MUNSHI A.: The opencl specification. In *Hot Chips 21 Symposium (HCS), 2009 IEEE* (2009), IEEE, pp. 1–314. 7

[RLS*ng] ROBERTS R., LARAMEE R. S., SMITH G. A., BROOKES P., D'CRUZE T.: Smart brushing for parallel coordinates. *IEEE Transactions on Visualization and Computer Graphics* (Forthcoming). 3

[RTL*16] ROBERTS R., TONG C., LARAMEE R., SMITH G. A., BROOKES P., D'CRUZE T.: Interactive analytical treemaps for visualisation of call centre data. In *Proceedings of the Conference on Smart Tools and Applications in Computer Graphics* (2016), Eurographics Association, pp. 109–117. 3

[Sca11] SCARPINO M.: *OpenCL in action: how to accelerate graphics and computations*. hgpu. org, 2011. 7

[SEP*15] SHI J., ERDEM E., PENG Y., WOODBRIDGE P., MASEK C.: Performance analysis and improvement of a typical telephone response system of va hospitals: A discrete event simulation study. *International Journal of Operations & Production Management 35*, 8 (2015), 1098–1124. 3

[SG18] SARIKAYA A., GLEICHER M.: Scatterplots: Tasks, data, and designs. *IEEE Transactions on Visualization and Computer Graphics 24*, 1 (2018), 402–412. 2

[SGC*15] SARIKAYA A., GLEICHER M., CHANG R., SCHEIDEGGER C., FISHER D., HEER J.: Using webgl as an interactive visualization medium: Our experience developing splatterjs. In *Proceedings of the Data Systems for Interactive Analysis Workshop (Oct. 2015), DSIA* (2015), vol. 15. 2

[Sha03] SHARP D.: *Call center operation: design, operation, and maintenance*. Elsevier, 2003. 3

[Shn96] SHNEIDERMAN B.: The eyes have it: a task by data type taxonomy for information visualizations. In *Proceedings 1996 IEEE Symposium on Visual Languages* (1996), pp. 336–343. 2, 4, 8

[The95] THE QT COMPANY: Qt application framework, 1995. URL: https://www.qt.io/. 4

[The16] THE KHRONOS GROUP INC.: Vulkan overview, 2016. URL: https://www.khronos.org/vulkan/. 8

[WBS07] WEINBERG J., BROWN L. D., STROUD J. R.: Bayesian forecasting of an inhomogeneous poisson process with applications to call center data. *Journal of the American Statistical Association 102*, 480 (2007), 1185–1198. 3