Visualizing Structural Artifacts in Long-read Transcriptomic Sequencing

Baraa Orabi (borabi@cs.ubc.ca) and Nico Ritschel (ritschel@cs.ubc.ca)

The University of British Columbia

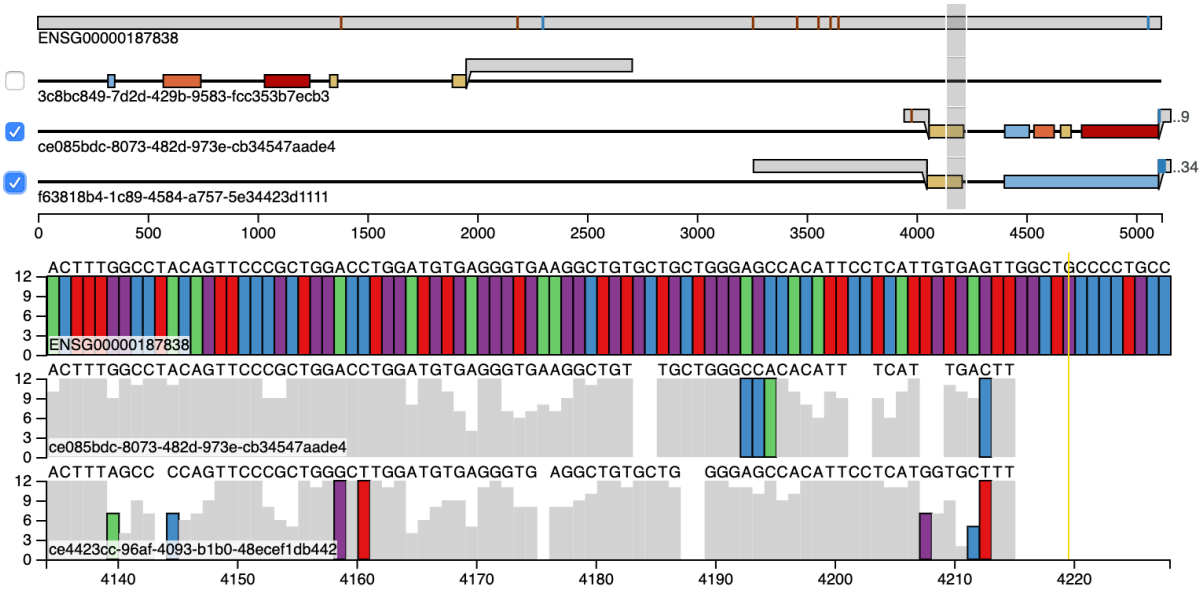

Fig. 1. Visualization of the PLSCR3 gene and 3 transcriptomic long-reads that are aligned to the gene (top half). A selected range of the gene and of 2 selected reads is shown in detail, allowing the inspection of the gene's base sequence, the sequencing quality of the reads and mismatches between the gene and reads (bottom half).

Abstract— Advances in DNA and RNA sequencing technologies have increased the length of transcriptomic reads sufficiently to cover full transcripts in a single read. However, tools for visualizing reads alignments have not adapted to this development and are primarily designed to visualize shorter reads. In addition, their chosen visualization idioms to not provide support for structural inter-read comparison, which is necessary for bioinformaticians to identify, compare and categorize transcript variations. This work presents a novel visualization design and implementation that supports bioinformaticians in inspecting and comparing long-read sequences for a selected gene. As interviews with domain experts show, the visualization's design with a high level of abstraction on a coarse level and precise selection tools for ranges of interest fits the needs for their daily work on transcriptomic data. Our solution is implemented in D3, Python, and Snakemake. Our code and a sample demo are publicly available on our GitHub repository at github.com/baraaorabi/LTR-vis/tree/code-freeze.

1 Introduction

Technological advances in biotechnology have made it easier than ever before for researchers to obtain large amounts of measurement data. While this development certainly had a positive overall impact on the efficiency of research and the quality of its results, it has also created a new challenge, which is to effectively handle and interpret the amounts of collected data. A field where this effect is particularly noticeable is that of genomics, where the ability to read longer DNA sequences at a time, combined with the reduced cost of doing so, has led to an explosion of the amount of available data.

While automated tool-chains provide fundamental support for bioinformaticians and biologists on a lower level of data processing, the task of interpreting and contextualizing measurements is still highly reliant on human expertise. To work effectively, it is essential for these experts to be able to quickly switch between a coarse overview of the available dataset and a fine-grained analysis of specific data ranges of interest. Modern visualization tools are available for many types of analyses, yet none of them is sufficiently suited for the analysis of long-read sequencing data to identify structural variations. This work aims to create a visualization that matches the specific needs of researchers working on this type of analysis task.

2 Background

Living organisms use DNA to encode genetic information. DNA molecules are chemical molecules and are composed of repeating sub-molecules known as *nucleotides (nt)*. There are four main types of nucleotides that are often represented with the letters **A**, **C**, **G**, and **T**. Thus, any DNA molecule can be abstractly thought of as a sequence written using a very limited alphabet containing only four letters. The collection of all DNA molecules in a cell is known as the genome. Genome sequences can become very long; for example, each human cell contains three billion nucleotides worth of DNA molecules.

Most of the functions in living organisms are performed by *proteins*. Proteins are molecules that have specific chemical structures es-

sential for their function. The structure of every protein is determined by a *gene*. Genes are specific substrings of the genome. The human genome contains about 20,000 genes which combined constitute about 2% of the whole-genome. Genes as isolated objects are often the level of abstraction that geneticist and molecular biologists deal with in their research.

To translate a gene to a protein, a copy of the gene called *transcript* is created. However, some parts of the transcript can be *spliced out*, meaning that they are removed before the transcript is translated into a protein. Different transcripts of the same gene can be spliced in different ways. Those alternatively spliced transcripts of the same gene are known as *isoforms*. Alternative splicing allows one gene to encode for multiple proteins with similar chemical structures but potentially completely different functions. The *transcriptome* is the collection of all isoforms sequences in a given cell. The human transcriptome contains about 80,000 isoforms, averaging four isoforms per gene. Thus, alternative splicing is the dominant reason for structural variation in transcriptomic data.

Sequencing is a technology that takes real DNA molecules and outputs digital files containing the nucleotide by nucleotide sequence of these different molecules. Each of record in these files is called a *read*; a read is the digital representation of DNA molecule. Ideally, a single DNA molecule is *sequenced* or read[1] end-to-end with no sequencing errors. However, reads generated by sequencing technologies are far from perfect. *Next-Generation Sequencing (NGS)*, the most commonly used class of sequencing today, generates short-reads with lengths ranging from 75nt to 250nt and has a sequencing error rate of 2 to 4 errors per 1000nt [10]. The reasonably low sequencing error rate of NGS reads makes NGS reads a powerful tool to detect single nucleotide mutations in a sample. On the other hand, the short length of NGS reads makes them difficult to utilize to identify larger structural mutations such as large deletions, insertions, and inversions.

Long-read Sequencing (LRS) is a newer class of DNA sequencing technologies. LRS read lengths can reach thousands to tens of thousands of nucleotides, orders of magnitudes more than NGS reads. However, LRS suffers from a very high sequencing error rate of 10-20% [11]. The high sequencing error makes using LRS to detect mutation of single nucleotides challenging, but its long read sizes make it a suitable tool for identifying structural variations.

Bioinformatics pipelines can automatically generate predictions of genetic relations based on reads, even in the presence of high sequencing error rates. For example, a tool for isoform discovery can use read alignments to predict the structure of thousands of gene isoforms that are present in the sample. Each of these isoform predictions is based on a small subset of reads that provide empirical support for the resulting isoform prediction. While automated predictions are an essential tool for researchers in the field, the results still require manual interpretation and filtering of a visibly clear false positive.

3 Related work

State-of-the-art visualization tools for sequencing data are designed for NGS short-read data. For example, *Integrative Genomics Viewer (IGV)* [9] and *Integrated Genomics Browswer (IGB)* [8] are desktop software packages that takes read alignment data alongside a target reference as input. Both IGV and IGB stacks as many short-reads in a single view as possible. They also restrict the base grid of the view to the reference coordinate grid since they assume that their input is NGS data which reads not long enough to be a possible alternative grid. Another popular genome alignment view is UCSC Genome Browser [4]. Genome Browser is a web application that is mainly focused on visualizing genomes alongside relevant genomic features such as genes, transposons, epigenetic marks, etc. Users may add custom tracks to Genome Browser that include their read alignment data which is laid out using a similar design to that of IGV or IGB.

All these tools are designed for short-reads and thus are good for characterizing single point mutation calls. However, LRS is often used to characterize larger structural variations that can span multiple genomic intervals. Circos plots [6] are one of the more commonly used design idioms to visualize some large structural variations. However, Circos plots are typically limited to genomic rearrangements and do not have an intuitive way to present the underlying reads that support the structural variation they visualize.

When using any of these tools or idiom with LRS data, two main limitations arise: 1) long-reads expressing variations can span two or more genomic intervals and 2) long-reads could contain large novel sequences that cannot be linearly represented on the reference coordinate grid.

[1] Rhymes with bed

Dataset	# of reads	Length (nt)		Throughput
		Mean	Median	(Gnt)
WGSC dRNA	10,302,647	906	677	∼10.0
WGSC cDNA	15,152,101	1,030	771	∼14.0
VPC cDNA	3,361,630	1,587	1229	∼3.3

Table 1. Statistical summary of the three LRS datasets.

4 Data

In this work we used data from transcriptomic LRS, reference genes and transcript sequences from the *ENSEMBL* database [3]. In addition, we derived additional data from the LRS data using *minimap2* genome alignment and mapping tool [7].

4.1 Transcriptomic LRS

We used three LRS datasets during the development of this work. The first two are publicly accessible datasets from Nanopore Whole-genome Sequencing Consortium (WGSC) [12]. Both WGSC dataset includes LRS of the transcriptome of NA12878, a well studied human cell line. However, one uses direct RNA (dRNA) based sequencing technology and while the other uses complementary DNA (cDNA) sequencing technology. dRNA LRS generates slightly longer, more accurate reads than cDNA LRS.

The third dataset is a private dataset from B. Orabi's research lab at the Vancouver Prostate Centre (VPC). The VPC LRS dataset is generated from the transcriptome of 22Rv1, a prostate cancer cell line and is generated using a cDNA approach.

LRS data is composed of records known as *reads*. A read has a read name that is, typically, a random string that is a unique identifier for the read in the dataset. The read also has a sequence which is a string of characters drawn from the alphabet: {**A**, **C**, **G**, **T**}. Table 1 summarizes the main read sequence statistics of the datasets.

Finally, each read has a string of quality scores. The quality string has the same length as the read sequence. The quality scores are encoded using the *Phred* quality scoring scheme [2]; a Phred score is a log-based likelihood of a single base being sequenced correctly and has an integer value between 1 and 60. A score of 60, the highest possible score, means a $1/10^6$ chance of being wrong. Quality scores, whether for isolated sequence positions or over an interval of adjacent positions, are helpful in dismissing some predictions, e.g., if some reads all support an insertion but have low quality for the insertion sequence then that might be dismissed as a mere systematic sequencing error.

4.2 Gene reference sequences and annotations

We used human gene reference sequences and annotations from ENSEMBL to generate target sequences and the derived data of read alignments on those target sequences. The reference includes over 20,000 protein-coding genes with over 80,0000 isoforms.

4.3 Read alignment

We used minimap2 for alignment and mapping. Minimap2 is a fast sequence aligner and mapper that utilizes a number of indexing techniques to achieve near-linear alignment (in terms of the query size). When aligning reads to reference gene sequences we will be using splice-aware alignment to ensure robustness against gene splicing.

5 Task Abstractions

Based on talks with domain experts and our own experience, we have defined the most frequently occurring tasks when these users work with long-read sequences.

5.1 Selecting genes and reads

A typical dataset can contain millions of reads that can be mapped onto thousands of genes. To allow users to gain any meaningful insights from this amount of data, they need to be able to select a single gene to focus on. When LRS technology is used for sequencing, each gene has on average a few hundred reads, from which either an automated tool or the user needs to select a subset that is relevant for the desired analysis task.

5.2 Observation and comparison of indels

The most common type of sequencing error that occurs in LRS are insertions and deletions of one or multiple nucleotides, often called *indels*. While existing bioinformatics tools can identify the positions of indels and handle them automatically when aligning reads, users should still be able to view indels to validate the alignment and judge the overall quality of a read. For this purpose, users need to be able to quickly infer indel sizes visually when examining and comparing reads, requiring potential filtering of minuscule indels based on a threshold size.

5.3 Contracting sparse regions

Genes often have large regions that are almost always spliced out when they are transcribed. Some genes can have lengths up to hundreds of thousands of nucleotides of which only a few thousand are expressed in transcripts. While the presence of spliced out regions is a relevant observation for experts, there is typically no need to display them true to their size in proportion to the overall gene. Contracting them, either automatically or manually, would allow experts to focus on the information that is actually relevant to them.

5.4 Identification of motifs of interest

Certain gene patterns that have a particular meaning to domain experts. An example is *poly-A tail* sequences that mark the end of a gene and that, if they are present in the middle of a read, indicate a sequencing error. Another example is the presence of adapter sequences of the library preparation stage which are expected to be present at the start of the read. Other, similar sequences that should be easily visible to experts.

5.5 Examining the nucleotide sequences and quality strings

When inspecting a small number of reads, users may be interested in viewing and comparing the concrete underlying base sequence of the reads and the gene. In addition, users may be particularly interested in the Phred quality scores of the sequence data.

5.6 Read-to-read comparison

One of the main targets of inspecting long-reads is the identification of structural variations. While using the annotated gene as a reference is ideal for identifying many of these variations, genomic rearrangements and variations could result in novel insertions of nucleotides that are present in reads but not in the annotated gene. The presence of the same novel insertion in multiple reads signals that such novel insertions are the result of true biological rather than the result of sequencing noise. Therefore, users might have an interest in not just comparing reads to the reference gene but also to use a read as the reference to compare others to it.

6 Solution

Based on our previously presented task abstractions, we have created a data processing pipeline and a visualization design with the goal of enabling users to execute most of these tasks as effectively as possible. The pipeline is implemented using Snakemake [5] and is available on our GitHub repository. Due to the limited scope of this manuscript, we have excluded the gene selection task, which constitutes its own separate data processing challenge. Instead, we assume that users have already pre-selected a single gene of interest and a moderately sized subset of less than 20 long-reads they consider relevant for them.

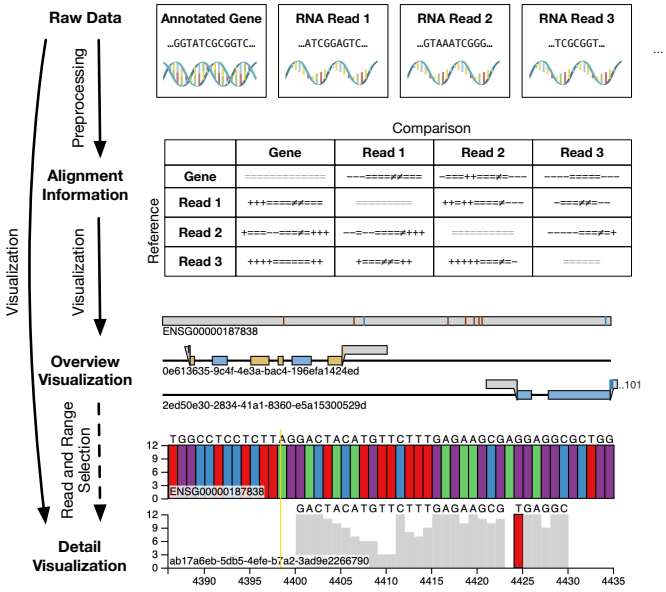

Fig. 2. Data processing pipeline used for generating the visualization.

6.1 General design considerations

All of the abstracted tasks we presented previously are either *Overview Tasks* which require the context of the entire gene or read, or *Detail Tasks* that focus on a localized range of the gene or read. As the length of a single gene can reach thousands to hundreds of thousands of bases, it is difficult to effectively support the whole range of described tasks in a single visualization. We have therefore decided to facet our visualization into an overview and a detail view.

Our overview and detail view use different levels of data abstraction: While the overview treats reads as sets of alignment fragments with certain alignment offsets and ranges in relation to the reference gene, the detail view has a substantially higher resolution and visualizes individual bases of the gene and reads. Figure 2 shows the resulting data processing pipeline to enable these views: The overview is exclusively based on pre-processed alignment data, while the detail view uses the alignments as well as the selected range in the overview to query the equivalent ranges in the raw data sequences.

While users may also be interested in limiting the selection of reads they are interested in, none of the identified tasks is limited to a single read. This means that an effective visualization always needs to be able to display the gene and one or multiple reads next to each other. Since this applies to both the Overview Tasks and Detail Tasks, we have decided to further facet each view into a Small Multiples visualization that shows reads and the gene next to each other. To effectively compare genetic bases, it is essential that they are aligned based on their position in the sequence. We have therefore decided to always synchronize this axis for each multiple of our respective views, as shown in Figures 1 and 2.

6.2 Read alignment overview

The read alignment view serves two fundamental purposes in our visualization:

1. It is the first view presented to users and therefore needs to give them an immediate summary of the overall structure of the visualized dataset consisting of both the gene and all reads with their corresponding indels.

2. It is the point at which all navigation interactions take place, allowing users to select the range of the genetic sequence they want to inspect in detail and to crop and zoom the view to focus on relevant areas of the gene.

We discuss how our visualization design serves these purposes in the following subsections.

6.2.1 Splicing alignment visualization

To summarize the whole dataset's structure, we have chosen alignment fragments as a data abstraction that retains an approximately constant size and complexity independent of the scale of the visualized gene. While the underlying length of a genetic sequence is frequently in the range of tens of thousands of data points, even the longest reads rarely have more than a dozen individual alignment fragments. Each fragment represents either a single region of alignment, insertion or deletion. As aligned regions and insertions have corresponding sequence data, we visualize them through filled blocks, while empty deletions are represented by a thin line. Since fragments of a single read cannot overlap, they can further be visualized very directly by using a single scale and axis for all fragments' sizes and offsets on the gene.

While aligned read fragments might have other data of interest associated with them that is relevant to the user, we have identified the quality of the alignment, which is defined by the relative amount of mismatches between the read and the reference, as the one that is most relevant to the user when comparing and evaluating indels. Consequently, we have decided to embed alignment quality directly into our overview by color-coding aligned fragments on a diverging scale.

The reference gene to which all reads are related is always completely available and spans the full available range for all read fragments. Therefore the only benefit of displaying the gene along the reads is consistency with the detail view, in which it is essential to display the reference gene's base sequence. To achieve this consistency while also providing further value to the user, we have decided to highlight motifs of interest through markers that are defined by a range on the gene as well as a categorical type that we decided to represent by its color.

While aligned fragments and deletions correspond to a specific range on the reference, this does not apply to insertion fragments, which therefore do not have representation on the used axis. For the task of comparing and evaluating indels, the most relevant information about insertions, besides their presence itself, is their size. To allow users to compare the size of insertions to that of aligned fragments, we have decided to encode both using the same scale. While insertions do not have a natural position on the used axis, they do have an order relative to the aligned blocks. In particular, most insertions are either *prefixes* that occur before the first aligned block, or *suffixes* that occur after the last aligned block. In addition to prefixes and suffixes, insertions can also occur between two aligned blocks, although this case occurs less frequently. When positioning insertions, we have therefore decided to place prefixes immediately before the first aligned block, suffixes immediately after the last aligned block and other insertions in the middle of the two surrounding aligned blocks.

To avoid occlusion between aligned blocks and insertions, and to clearly distinguish them from each other, we have decided to place insertions on a separate row above all aligned blocks and deletions. While this can still lead to occlusion between multiple insertions in rare cases, it ensures that aligned fragments always remain fully visible. Since alignment quality does not apply to insertions, we have decided to not color-code them but instead use the same highlighting of motifs of interest as for the reference gene.

Our chosen positioning approach can cause prefixes and suffixes to extend beyond the range of the gene axis. To avoid misinterpretation, we cut off these overlong ranges but place a number at the edge of the axis that represents the length of the cut-off range.

6.2.2 Navigation

The most fundamental navigation interaction our overview provides is the selection of a range that defines the boundaries for the detail

Fig. 3. Block-based visualization of a read's alignment. Block positions and sizes correspond to their offset on the gene and length. A diverging color scheme represents alignment quality. Insertions with no alignment to the gene are shown in gray, with categorically color-coded motifs of interest. If insertions extend beyond the canvas, they are cut off and the remaining length is shown as a numerical label.

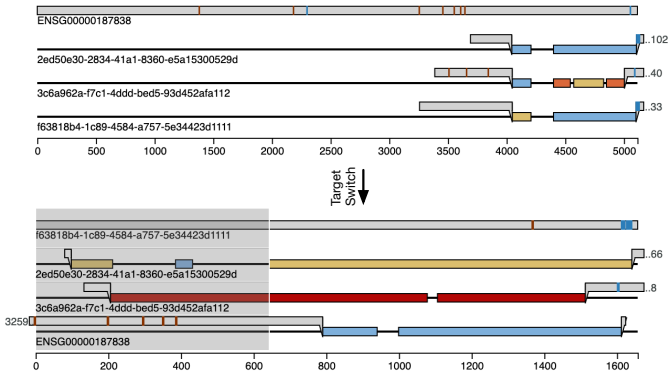

Fig. 4. Target switching from an alignment to the gene to an alignment to a selected read. The switch can reveal relevant read-to-read alignments, such as those in the selected area, that would not be visible otherwise.

view. This interaction is illustrated in Figure 1. Following our design principle of synchronized axes, range selections always apply to all reads and the gene at once. In addition to selecting a range on the data sequence, users can also select a subset of reads that they intend to compare to the reference gene. This selection is particularly important to improve the visibility of the relevant data by reducing the number of small multiples that are displayed in the detail view. Figure 1 shows the selection of reads through check-boxes.

In addition to selection interactions, the overview visualization also supports advanced navigation through cropping and zooming the displayed data range. Unlike selection, these interactions change the visible data for both the overview and the detail view. Figure 5 illustrates the two interactions: Through a keyboard shortcut, a selected range of data can be cropped out, resulting in the remaining data to be stretched to fill the empty space, or zoomed in, effectively cropping out the remaining data ranges. To enable the most flexible navigation and consistent user experience, both cropping and zooming can be repeated and applied in any order.

The ability to hide irrelevant data can greatly improve the visibility of relevant data ranges, as particularly the right example in Figure 5 illustrates. However, it also introduces perceptual complexity as the data axis can become discontinuous like on the left side of Figure 5. To ensure that users are always aware of discontinuities in their data, we add a black marker wherever discontinuities occur.

As insertions do not have a fixed position on the reference gene, they are excluded from cropping unless the position they are visually anchored to is cropped out. This can lead to them spanning across discontinuities as shown on the left side of Figure 5. While this result might not seem intuitive at first, we considered alternatives and concluded that the chosen approach, once rationalized, is the most useful one in practice and causes the least perceptual distortion of the underlying data.

6.2.3 Target switching

Besides the previously presented navigation interactions, the task of read-to-read comparison requires an additional form of overview navigation: Instead of using the annotated gene as a target to align reads to,

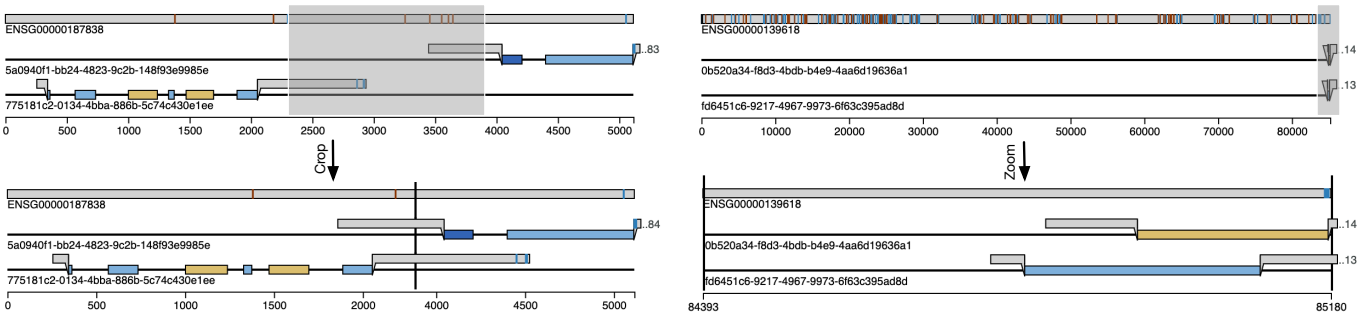

Fig. 5. Advanced overview navigation by cropping out or zooming into the selected range. Both types of operations can be combined in any order to precisely customize the visible data range. If a selection spans one or more cuts, the corresponding data is also cut from the detail view.

users may want to explore alignments between individual reads. Using a read as an alignment reference is particularly useful for comparing insertions between reads, which is not enabled by the previously presented overview.

The overview provides users with two ways to trigger a target switch. By clicking on a read's name, this read is selected as the target and the remaining reads are aligned to it. Alternatively, users can directly double-click on an insertion fragment to not only switch the target to the corresponding read but also highlight the area of the read covered by the insertion. This provides users with additional context to simplify a before-after comparison.

While alignment fragments for different targets are derived from the same raw base data and often related, there is not necessarily a full symmetry between, for example, aligning a read to the gene or the gene to a read. This means that target switching not just provides a different lens on the same dataset but can provide access to additional derived information not accessible otherwise.

Figure 4 illustrates a target switch from the default read-to-gene alignment to an alternative alignment to a selected read. Besides revealing potentially relevant motifs of interest on the read, target switching also allows easy analysis of inter-read alignments. In the top view of Figure 4, it is not clear if the prefixes have any relation. After target switching, the bottom view shows that some of the insertions in the selected range are related. This may support a user in distinguishing measurement errors from real biologically relevant findings.

6.3 Base sequence detail view

After users selected a sequence range and relevant reads to display in the overview, the base sequence detail view allows them to examine this data in its full resolution. For a given range of the currently selected target, the detail view displays two attributes for each read and sequence position: The called base for a single nucleotide is displayed as a color-coded categorical attribute, and the Phred quality score is height-coded, allowing the detail view to be read like a bar chart. Instead of an explicit legend for the called base colors, the detail view displays the assigned letters on top of the view as long as the selected range is reasonably small. While our chosen color scheme is colorblind-friendly, this redundant encoding intends to make the displayed information even more accessible.

The ideal length of a selection to be effectively displayed and examined in the detail view is in the range of dozens but not hundreds of data points per read. Figure 6 shows the detail view for such an amount of data. Since the primary purpose of the detail view is to enable the comparison of reads to the target and particularly the inspection of mismatches, we have decided to only color-code bases on the reads that differ from the target. This not only serves to avoid an otherwise high level of redundancy in the display but also explicitly highlights mismatches to the user. This highlighting remains effective even for larger selection ranges, such as hundreds or even up to 1500 bases (which is the maximum range that the detail view supports for performance reasons). Figure 7 shows such a large range, which can

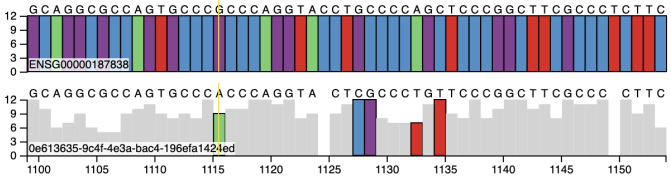

Fig. 6. Detail view with color-coded and labeled individual bases. For each base, bar height represents the sequencing quality. For reads, only bases that do not match the gene are colored to highlight them.

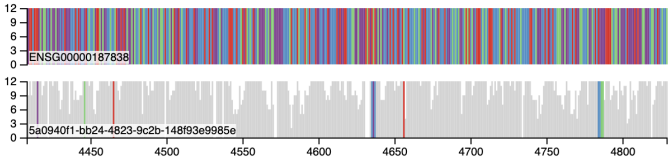

Fig. 7. While individual bases become indistinguishable for larger ranges, mismatches highlighted by color remain visible and can allow a more fine-grained identification of areas of interest than the overview.

be useful for users to identify areas that contain mismatches of interest and zoom in further.

While our overview allows the selection of a subset of reads to be displayed in the detail view, it can be difficult to accurately compare individual positions across the different tracks of the small multiples view. To improve this accuracy, the detail view provides a vertical line marker that spans all tracks and follows the mouse cursor as the user hovers over the data. While the name of the displayed read or gene is displayed for each track, we also ensure that the order of the reads in the overview is always consistent with the order of the detail view, even if reads are added out of order.

7 Implementation

To implement our visualization, we created a two-stage pipeline: To pre-process raw sequence data from reads and the annotated gene, we used the existing sequence alignment tool *minimap 2* [7]. Using this data, we then used the D3.js framework [1] to visualize both the alignment information as well as the raw sequence data.

While the preprocessing of the alignments can be performed in less than a minute on a desktop computer, for technical reasons we decided to pre-compute all possible read-to-gene and read-to-read alignments for each gene. The resulting data size for one gene and up to 15 reads was in the range of tens of MB which is reasonable this use case. It may however not scale sufficiently to allow pre-building a full database for a more substantial dataset as the ones shown in Table 1. Besides an increased implementation effort and a potentially longer delay when loading new data, we do not foresee major challenges in switching to on-demand computation for future implementation.

Due to the chosen abstraction for our overview, we had no issues in drawing even exceptionally long genes with hundreds of thousands of nucleotides. For the detail view that shows the genetic sequences in their full resolution, the performance for re-rendering larger selections dropped notably enough to cause usability issues when selecting ranges. We therefore decided to limit the maximum selection size to be displayed to 1500 bases, which led to acceptable performance even on machines with limited computing power. We consider this limitation reasonable since the detail view becomes visually too cluttered for larger selections and the overview provides navigation features that allow users to fully customize the displayed range to their needs.

8 Evaluation

We conducted interviews with four domain experts in computational biology or bioinformatics algorithm development: E1, E2, E3 and E4. The interviews were loosely structured and each lasted about 15 minutes. Each interview started by introducing the visualization on two already preprocessed genes, PLSCR3 (length ∼5,000 nt) and BRCA1 (length ∼125,000 nt). We choose these two genes to present the scalability of the solution with respect to the gene length. Each gene visualization included about 20 reads from the WGSC cDNA (Table 1).

All interviewees are researchers at the Vancouver Prostate Centre. E1 is a senior computer science PhD student working on genome assembly and structural variation detection problems using LRS data. E2 is a computer science PhD student working transcriptomic LRS data to detect gene fusions. E3 is a bioinformatics PhD student working on single-point mutation calling using NGS sequencing and ChIP-seq data, a type of sequencing that correlates far regions of the genome based on their interaction with a given protein. E4 is a principal investigator whose research interests focus on bioinformatics and sequencing algorithms development.

After introducing the main interactivity features and mark encodings, our interview focused on answering the following questions:

1. What was done right and how does it help in your work?
2. What was done wrong and needs to be modified?
3. What features are missing and need to be added to the visualization?

The overall feedback we received from the interviews was positive: Interviewees found the visualization marks intuitive and that most idioms serve their task needs. All the interviewees pointed out that cropping of sparse regions is a very helpful feature for their workflow. E1 and E2 mentioned that cropping would enable them to visualize variations caused by large deletions or inversions and that are typically thousands to hundreds of thousands of nucleotides long which other genome alignment viewers (IGV and Genome Browser) such cannot visualize effectively. Additionally, all interviewees found scalability of the visualization to large targets is a major advantage over present alignment viewers. E3 pointed out that the choice to set the color of matching characters in the detail view to grey and only coloring mismatching bases was excellent to quickly locate mismatches. E3 immediately pointed out that the fusion of quality mark (height of the bar) and mismatch mark (fill color of the bar) makes it easy for them to identify likely false positive mutations. Additionally, E3 found that target switching using the insertion marks and seeing that other reads have aligned blocks on the selected target read insertion motivates them to consider that those insertions are not random noise.

Interviewers criticized some design choices in the visualization. E1 and E3 found it difficult to track horizontally match blocks on the side of the canvas to their respective checkboxes on the other end side. E3 suggested that we use alternating background highlighting to facilitate visual tracking of read blocks. E3 also pointed out that the overlap in the choice of colors for the motifs and match blocks is, at least initially, confusing. E4 mentioned that the space occupied by the read names would be better utilized if read names were hidden and more reads are packed vertically. E4 suggested that displaying read info upon mouse hover would suffice for most of their needs. E1 mentioned that tracking different read alignment intervals when target switching from the gene to a read is important and that the current target switching mechanism does not facilitate that.

All interviewees mentioned a number of features that they think the visualization needs to be extended with. E1 and E4 mentioned that starting from a whole-genome view would be more intuitive to typical users and would enable certain analyses that the current visualization does not provide. Such analyses include detecting variations which span extends beyond the boundaries of a single gene. On a similar note, E2 and E4 stressed the need to visualize the secondary alignments of long-reads since those alignments are important to consider when evaluating gene fusion or structural variation events. E3 stressed the need to visualize small insertions and deletions that occur within a match block in the detail view and that the current detail view does not allow the user to easily distinguish between small deletions and skipped (intronic) regions that fall outside match blocks. Finally, E1 stressed the importance of allowing the user to sort and filter the reads during the visualization stage and not to restrict it only to the preprocessing stage.

9 Discussion

The results of our evaluation confirm that our visualization design provides substantial benefits over existing tools. Some of these, like the chosen approach for small multiples faceting or the ability to switch alignment targets, are specifically useful for the typical tasks executed on long-reads. Particularly the chosen approach to iterative cropping and zooming might however also be applicable and effective for tools that visualize short-reads or other genetic sequence data.

In the following, we discuss the limitations of the presented work and outline intended future work.

9.1 Limitations

The most significant factor that limits the practical application of our tool is that it does not address the task of selecting genes and reads. This means that biologists need to use other tools to select genes and particularly reads of interest from a dataset. While our overview visualization would be helpful for selecting reads, it does not scale sufficiently enough to be used for inspecting more than a few dozen reads at once. We could see a more aggressive scaling approach with a matrix-like alignment of read overviews as an interesting starting point for future work, but expect additional modifications and a potentially more coarse abstraction level to be necessary.

Another limitation of our visualization is that it assumes that reads will 1) span a single gene and 2) have a single alignment. While this assumption hold for most reads, many structural variations of interests are inferred from reads that map to more than one gene and/or have multiple alignments.

Finally, a limitation that is the result of the chosen implementation frameworks is the inability to save and restore the state of the visualization or export data. For similar reasons our tool currently misses the ability to add annotations and notes to data for later use. While most visualizations for biologists are stand-alone desktop applications, our current tool is implemented as a web-based view. This does not generally rule out the option to save and load local files, but may result in a less convenient work-flow for users.

9.2 Future work

We believe that many of the limitations we have described as well the lacking features highlighted by our user can be overcome by future work. Adapting our current overview visualization for this task or developing a new design from scratch would be an interesting challenge.

In addition, we are interested in conducting a more structured and extensive evaluation of our work. While we have provided participants with sample data when we demonstrated our tool, a more realistic test of its practical usefulness would be to let users work on their own datasets and execute realistic every-day tasks. Interviewing them about their success and satisfaction could provide significantly more useful insights into missing features or other room for improvement.

10 Conclusion

This work presented a novel visualization design and implementation to support bioinformaticians when working on long-read sequences. While our design does not yet support the full top-down work-flow of biologists, it provides an effective approach to inspecting transcriptomic read alignments and the underlying genetic base sequence. We expect this work to be a foundation for the future development of a full-scale environment for long-read selection and inspection.

References

[1] M. Bostock, V. Ogievetsky, and J. Heer. D^3 data-driven documents. *IEEE transactions on visualization and computer graphics*, 17(12):2301–2309, 2011.

[2] B. Ewing, L. Hillier, M. C. Wendl, and P. Green. Base-calling of automated sequencer traces using phred. i. accuracy assessment. *Genome research*, 8(3):175–185, 1998.

[3] S. E. Hunt, W. McLaren, L. Gil, A. Thormann, H. Schuilenburg, D. Sheppard, A. Parton, I. M. Armean, S. J. Trevanion, P. Flicek, et al. Ensembl variation resources. *Database*, 2018, 2018.

[4] W. J. Kent, C. W. Sugnet, T. S. Furey, K. M. Roskin, T. H. Pringle, A. M. Zahler, and D. Haussler. The human genome browser at ucsc. *Genome research*, 12(6):996–1006, 2002.

[5] J. Köster and S. Rahmann. Snakemake—a scalable bioinformatics workflow engine. *Bioinformatics*, 28(19):2520–2522, 2012.

[6] M. Krzywinski, J. Schein, I. Birol, J. Connors, R. Gascoyne, D. Horsman, S. J. Jones, and M. A. Marra. Circos: an information aesthetic for comparative genomics. *Genome research*, 19(9):1639–1645, 2009.

[7] H. Li. Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics*, 34(18):3094–3100, 2018.

[8] J. W. Nicol, G. A. Helt, S. G. Blanchard Jr, A. Raja, and A. E. Loraine. The integrated genome browser: free software for distribution and exploration of genome-scale datasets. *Bioinformatics*, 25(20):2730–2731, 2009.

[9] J. T. Robinson, H. Thorvaldsdóttir, W. Winckler, M. Guttman, E. S. Lander, G. Getz, and J. P. Mesirov. Integrative genomics viewer. *Nature biotechnology*, 29(1):24, 2011.

[10] M. Schirmer, R. D'Amore, U. Z. Ijaz, N. Hall, and C. Quince. Illumina error profiles: resolving fine-scale variation in metagenomic sequencing data. *BMC bioinformatics*, 17(1):125, 2016.

[11] R. R. Wick, L. M. Judd, and K. E. Holt. Performance of neural network basecalling tools for oxford nanopore sequencing. *Genome Biology*, 20(1):129, 2019.

[12] R. E. Workman, A. Tang, P. S. Tang, M. Jain, J. R. Tyson, P. C. Zuzarte, T. Gilpatrick, R. Razaghi, J. Quick, N. Sadowski, et al. Nanopore native rna sequencing of a human poly (a) transcriptome. *BioRxiv*, page 459529, 2018.