Mito-AssemblyVis: Mitochondrial Genome Assembly Assessment Visualization

Armaghan Sarvar (armaghansarvar@gmail.com), Cecilia Yang (ceciliayang1276@gmail.com)

Nov 15, 2022

Introduction

Mitochondrial genomes (mitogenomes) are short and circular DNAs in cellular organelles called mitochondria. Mitogenomes are significantly more abundant than nuclear genomes per cell and are less prone to degradation thereby allowing forensic and environmental scientists as well as anthropologists to obtain DNAs from samples that consist of little biological material [1]. In the past, the high sequencing cost and the inaccessibility of DNA sequencing technologies led to a shortage of complete mitogenomes of some culturally and environmentally important animal species in public databases such as the National Center for Biotechnology Information (NCBI). Nowadays, whole genome sequencing technologies have become much cheaper and more easily accessible, making it possible to assemble mitogenomes on a large scale. To ensure the accuracy of the assembled mitochondrial DNA sequences, choosing effective visualization methods for the quality assessment of mitogenome assemblies is very crucial in the field.

During the past year of Cecilia's graduate studies, she has focused on creating a mitogenome assembly pipeline (mtGasp) that takes in short sequenced DNA reads as input and generates a reconstructed mitogenome as output. Visualization tools can be applied to provide insights into the accuracy of the reconstructed mitogenome which will eventually allow us to assess the performance of her pipeline compared to other published mitogenome assembly pipelines. The key attributes we intend to visualize include sequence length, genome coverage, quality metrics (e.g., number of gaps and misassemblies), mapped regions between two or more sequences, and genome annotation results (the orientation and location of the genes). Mitogenomes are approximately 16,000~20,000 base-pair long, so sequence length will help us evaluate the completeness of the assembled sequence. Coverage indicates the number of DNA reads mapped to a given nucleotide in a reconstructed sequence [2]. Higher genome coverage is often associated with better genome assemblies, meaning the resulting reconstructed genome is close to its original form. Quality metrics reflect the overall accuracy of the assembled genome. Genome annotation is a method that determines the specific gene locations. Potential genomic rearrangements and the completeness of assembled mitogenomes can be investigated by visualizing the mapped regions between the reconstructed and a reference genome and comparing their annotation results. We propose a web visualization interface that implements the published visualization packages/tools in the back-end and allows users to upload their genome assembly data and visualize the genome assembly results based on their specified design choices (circos plot, parallel coordinate plot, annotation plot). This proposed web interface not only offers a one-click solution to visualize the mitogenome assembly results but also facilitates the comparisons between the performance of different assembly pipelines. In this work, the assembly results of Cecilia's novel pipeline (mtGasp) and two other state-of-the-art assembly pipelines will be used to generate our visualization prototype.

Related Work

Gosling [3] is a grammar for visualizing genomics data that is accompanied by the toolkit Gosling.js and provides multi-scale genomics visualizations. The JavaScript toolkit is built on an available platform for data visualization in a

web-based manner to simplify the visualizations of different genomics data formats.

Considering related tools that look into contigs or chromosomes in a fine-grained manner, we could mention the Integrative Genomics Viewer (IGV) [4], which is an interactive linear genome browser tool that supports the integration of the common types of genomic data, investigator-generated or from publicly available sources. In the context of our project, it is particularly useful when the user needs to investigate the reads aligned to an assembly, to derive information such as the coverage. An improvement over IGV is the Ribbon tool [5] which provides better support for visualizing long-read sequencing data, showing how alignments are positioned within both the reference and read contexts. This gives an intuitive view that enables a better understanding of structural variants and the read evidence supporting them.

Icarus [6] is a visualization tool that looks more specifically into genome assemblies. Icarus is an open-source software integrated with QUAST (A quality assessment tool for genome assemblies) [7] and complements its output with interactive visualizations of assembly alignments to the reference genome, and various features detected by QUAST (e.g. misassemblies).

Considering looking at the assembly aligned to the reference genome, there is also Xmatchview [8] that compares pairs of genomes to each other. Although it has been used to compare chloroplast assemblies to an NCBI reference, it does not handle multi-fasta sequences; hence, one could use it to compare two scaffolds in the same genomic region, but not multi-chromosome.

Similar to the above, with the assumption of having access to the reference genome, another group of algorithms can detect large genomic rearrangements using Circos [9] for their visualization step. Circos, is a Perl-based visualization software that requires setting up configuration files with a large number of parameters. R packages like RCircos [10] or ggbio [11] provide functions to display circular plots for genomic data. However, these tools may not filter specific genomic inputs.

HiPlot [12] is a python package that facilitates multi-dimensional data visualization, this tool was initially designed for deep-learning hyperparameter tune-ups. We proposed an alternative implementation for this tool - displaying the assembly results generated by different sets of assembly parameters which help potential users tune up their assembly pipelines.

MitoZ-Visualize is a subfeature of the MitoZ [13] which serves as a toolkit for animal mitogenome assembly, annotation, and visualization. When the “visualize” subcommand is activated, MitoZ produces a circular genome annotation plot displaying gene names, locations, orientations, and genome coverage.

pyGenomeViz [14] is a matplotlib-based python package that facilitates genome sequence similarity and gene feature comparisons. The output figure can be saved in different formats (e.g., HTML, PDF, PNG).

Assessment of mitogenome assemblies gauges both the completeness and contiguity of an assembly and helps provide confidence in downstream biological insights. Although the visualization tools discussed above have one or more features suitable for mitogenome assembly evaluation visualization, there is currently a lack of visualization toolkits that are specifically designed for mitochondrial genome assembly evaluations.

Data Abstraction

For data generation, our novel pipeline (mtGasp) and two state-of-the-art mitochondrial genome assembly tools called GetOrganelle [15] and MitoZ [13] were used to assemble mitogenomes from whole genome sequencing data. The target species for this prototype work is southern sea otter which is a culturally and environmentally important animal species in Canada, we downloaded the DNA sequencing data of a southern sea otter sample (SRR8597300) and a mitogenome reference sequence (accession number NC_009692.1) from NCBI.

Table 1. Attributes in parallel coordinates plot (HiPlot) in **Figure 1**.

Variable	Type	Description	Possible Values
K-mer Size (bp)	Quantitative	K-mer size used in assembly	1 to 150 (150 is the DNA read length)
Pipeline Name	Categorical	Name of the assembly pipeline	3 possible values: mtGasp, MitoZ and GetOrganelle
Number of Gaps	Quantitative	Total number of gaps found in a given sequence	0 to sequence length (e.g., 16500 possible values if the mitogenome sequence length equals to 16500 bp)
Number of Misssemblies		Total number of misassemblies found in a given sequence	
Average Gap Size (bp)		The average number of nucleotide base pairs (bp) of the gaps found in a given sequence	
Number of Sequences		The total number of sequences in the assembly output fasta file. (Fasta file is a text-based file storing nucleotide or amino acid sequences)	0 to infinity (Although mitogenomes are expected to be 16000bp~20000bp long, the sequence length can be much longer than expected when there's an assembly error)
Sequence Length (bp)		The total number of base pairs in a given sequence	
Genome Fraction (%)		Total number of aligned nucleotide bases in the reference divided by the size of a given genome	0-100

Table 2. Attributes in Genome Annotation Plot in **Figure 2.**

Variable	Type	Description	Possible Values
Sequence Type	Categorical	The type of the input sequence	4 possible values: sequences generated by 3 pipelines, reference sequence
Gene Start Position	Quantitative	Start nucleotide position of a gene	0 to infinity (Although mitogenomes are expected to be 16000bp~20000bp long, the sequence length can be much longer than expected when there's an assembly error)
Gene End Position		End nucleotide position of a gene	
Gene Orientation	Categorical	The orientation of the gene	2 possible values: forward and reverse
Genome Coverage	Quantitative	The number of DNA reads mapped to a given nucleotide in a reconstructed sequence	0 to infinity

Table 3. Attributes in Circos Plot (GGisy) in **Figure 3**.

Variable	Type	Description	Possible Values
Sequence Type	Categorical	The type of the input sequence	4 possible values: sequences generated by 3 pipelines, reference sequence
Sequence Length (bp)	Quantitative	The total number of base pairs in a given sequence	0 to infinity (Although mitogenomes are expected to be 16000bp~20000bp long, the sequence length can be much longer than expected when there's an assembly error)
Strand Match	Categorical	The orientation of the matched sequence	2 possible values: forward and reverse
Identity Percentage (%)	Quantitative	The level of identity between the query sequence and reference sequence	0-100

Task Abstraction

To compare quality across multiple assemblies, a set of standard metrics are typically calculated and then compared to one or more gold-standard reference genomes. While several tools exist for calculating individual metrics, applications showing comprehensive evaluations of multiple assembly features are, perhaps surprisingly, lacking. Our visualization toolkit will allow a bioinformatician to assess a genome assembly algorithm and compare its output with other assembly tools. The levels of this assessment are as follows:

At the assembly level, bioinformaticians often need to fine-tune the assembly toolbox parameters to get the best-assembled genomic sequence based on their pre-determined evaluation criteria. Hence, using a parallel coordinates plot (Figure 1) for fine-tuning the hyper-parameters would be advantageous.

At the analysis level, the bioinformatician would want to know how identical the assembled output is compared to a ground-truth reference genomic sequence using a circos plot (Figure 3) and whether a complete set of genes can be found on the reconstructed sequence (Figure 2), indicating how complete the results of different assembly pipelines are. Also, the user might need extra information, such as the coverage levels, which give more insight into how accurate the assembly is since higher coverage means one can be more confident with the assembly output in a specific region (Figure 2).

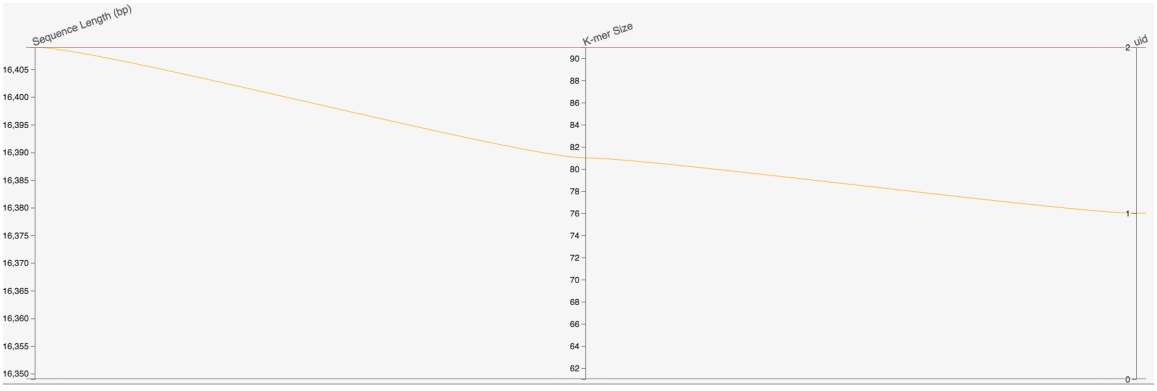
Once the user is able to get the optimal parameters for one assembly pipeline and analyze the assembly output quality, they may also want to compare sequences generated by different Mitogenome assembly pipelines (Figure 2 and Figure 3).

Solution

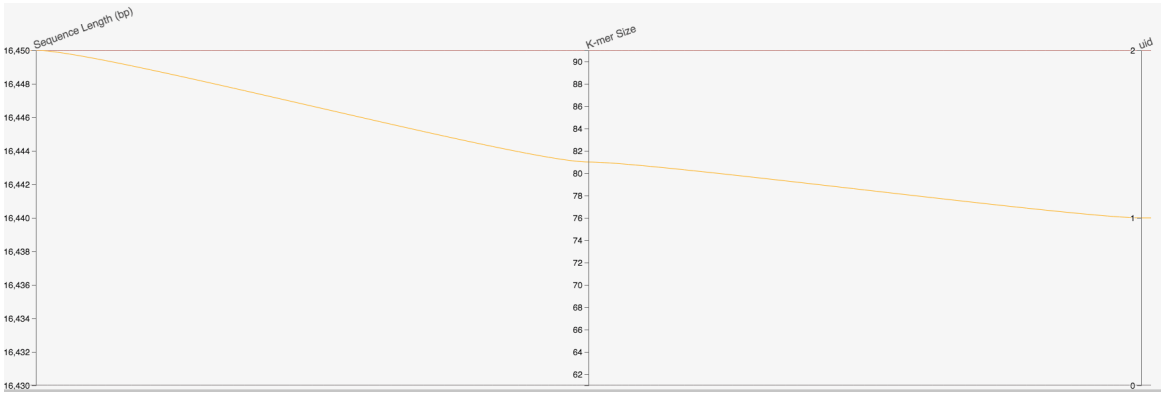
Idiom 1 - HiPlots for Hyperparameter Tuning

As mentioned in the related works, HiPlot (Figure 1) is an interactive visualization tool that, given experimental data points, helps us understand which parameters influence the metrics we want to optimize. However, this tool has never been used in the task of genome assembly parameter optimization. The HiPlot tool can be used to illustrate the relationships between different pipeline parameters, such as the k-mer sizes (61, 81, 91), and the final assembly results (e.g., sequence length, genome coverage) using parallel coordinate plots. This multi-dimensional data visual will help bioinformaticians to tune up the mitogenome assembly pipeline to obtain optimal results using the best assembly parameters.

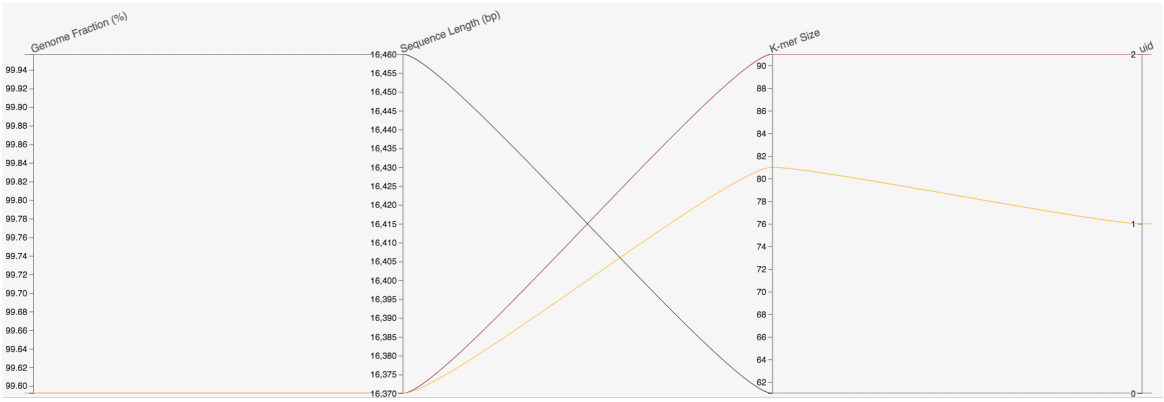
As shown in Figure 1, the majority of the attributes in Table 1 are not shown in the parallel coordinates plot, this is because HiPlot automatically filters out the attributes (e.g., Average Gap Size, Number of Gaps) that do not have unique values thereby allowing users to pay more attention to the key metrics that help select the optimal mitogenome assembly resulted from the 3 k-mer settings (61, 81, 91). As a result, the optimal k-mer sizes for mtGasp, GetOrganelle, and MitoZ are 91, 91, and 61, respectively.



(A) mtGasp



(B) GetOragenelle



(C) MitoZ

Figure 1: HiPlot [12]: Multi-dimensional Visualization of the Quality Metrics from the assemblies generated by various sets of k-mer sizes. Please note that HiPlot only displays the attributes (e.g., sequence length, genome coverage) that have a distinct value from each k-mer assembly run. As a result, many attributes listed in Table 1 are omitted in the final HiPlot visualizations. (A): The result for mtGasp pipeline. (B): The result for GetOrganelle pipeline. (C): The result for MitoZ pipeline.

Idiom 2 - Genome Annotation Visualization

In a complete genome assembly pipeline, genome annotation is conducted on the result of the assembly step, which is a process of inferring the structure and function of the assembled sequences. Here, we propose a circular genome annotation visualization such as a Chord diagram, where the results of the different assembly pipelines are shown in a circular representation to provide a quick and easy way to spot global patterns, features, or regions of interest based on

the mitogenome annotation results. More specifically, as shown in Figure 2, information such as the mitochondrial gene names, strands they are located on, and their starting and ending positions are visualized at the genomic scale to aid in understanding the organization of a genome or the similarities and differences across genomes resulting from different assembly algorithms. Other information, such as the coverage of the base pairs, will also be integrated into the visualization to help us compare the completeness of the results. This way, these algorithms can be easily compared to each other regarding the final gene positions.

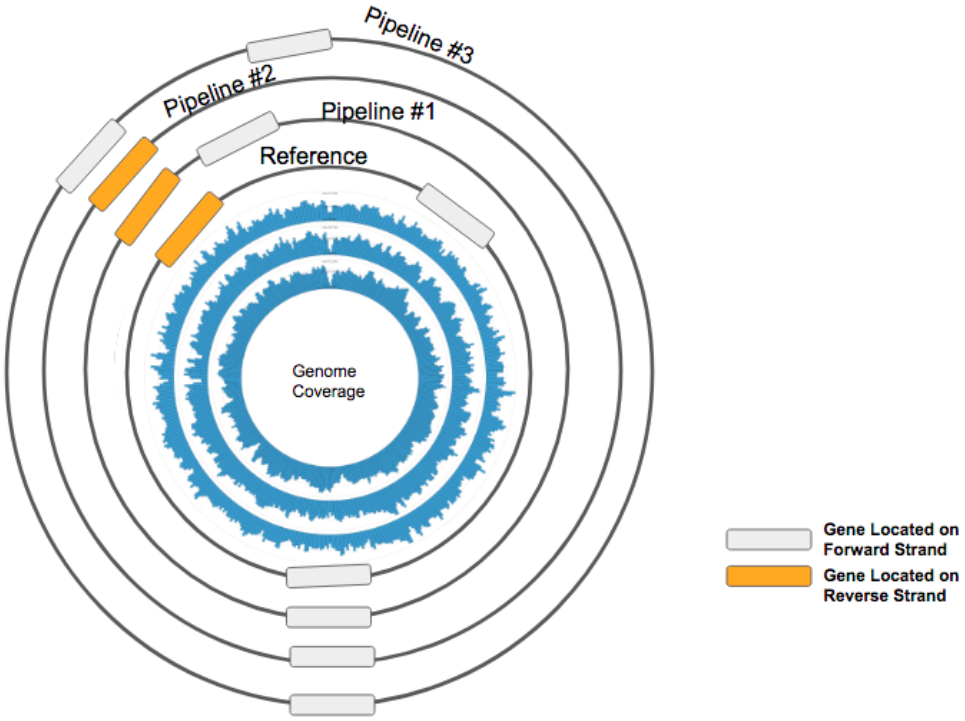


Figure 2: Multi-Layer Genome Annotation Plot: The 4 outer layers (from innermost to outermost) are the reference mitogenome and 3 pipeline-constructed sequences. The size of the annotated genes and their locations relative to the reference genome are also shown. Genes found on different strands (forward, reverse) are indicated by two hues (grey, orange). By comparing the annotated genes on reconstructed genomes with the reference, missing genes or gene relocations can be observed. The 3 inner blue layers (from innermost to outermost) are genome coverage from pipelines #1, #2, and #3, respectively.

Idiom 3 - GGisy-Based Circos Plot

We propose a GGisy-based module for generating a Circos-based genome assembly consistency plot given a set of contigs relative to the reference genome. This pipeline visualizes large-scale translocations or misassemblies in draft assemblies, but it can also be useful when trying to show synteny between whole genomes. Given only a reference genome fasta file and an assembly scaffolds fasta file, this analysis is good for getting a quick qualitative view of the misassemblies in a genome assembly. One possible drawback of this method is that small misassemblies, possibly mediated by repeats, might not be visible. In Figure 3, we can see the resulting visualization for the assembly data chosen in Table 3.

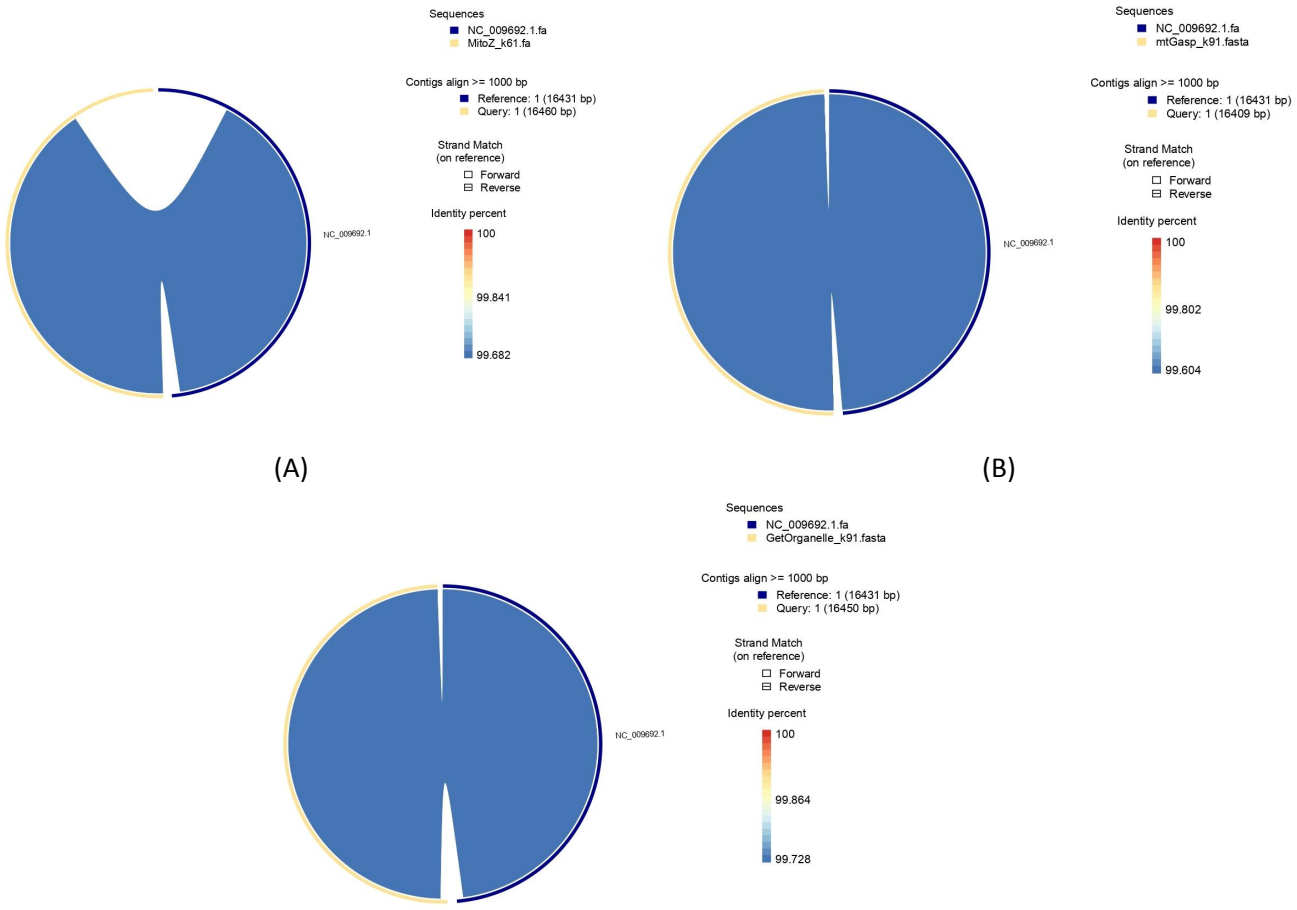


Figure 3: GGisy-Based Genome Assembly Consistency Plot: Exploration of the relationships between reference genome sequence (right) and pipeline-reconstructed sequence (left). Regions with at least 50 percent similarity are highlighted by the lines within the circle. The color saturation reflects the similarity levels. (A): The result for the MitoZ pipeline, (B): The result for the novel mtGasp pipeline, (C): The result for the novel GetOrganelle pipeline.

Implementation

The first phase of the project was generating the input data which consisted of the following steps: 1) Extracting target species sequencing reads from possible databases, 2) Generating the assemblies using the aforementioned pipelines, and 3) Parsing the resulting assembly outputs and the data annotation files. The Python programming language and the Bash Unix shell and command language were used for the above steps. As for the visualizations, so far, we have implemented Mito-AssemblyVis using Python libraries such as HiPlot [12] and GGisy [18]. In the future and more specifically, for implementing idiom 2 of the solution, libraries such as pyCircos [19], and Mummer2Circos [20] will most probably be utilized. We intend to investigate using the Flask [21] web application back-end framework to generate a dashboard for user interaction and show the final results. Before that, the front-end module of this application will be implemented using HTML and CSS to structure the web page and its content.

Usage Scenarios

Scenario 1:
Another user might need to find out the optimal set of parameters used in either of the assembly pipelines that lead to the best result. Here, Idiom 1 can be utilized. After running their assembly pipelines with the hyper-parameters to be evaluated, the user will generate and upload a tabular dataset that summarizes the parameters and the quality metrics. For example, considering mitogenome assemblies, most often, sequence length along with the number of misassemblies or gaps are evaluated. For our novel assembly pipeline, hyper-parameters such as different k-mer sizes help users to select the optimal assembly based on the above quality metrics. Depending on the assembly pipeline, the target hyper-parameters can change.

Scenario 2:
Imagine a bioinformatician or clinical user has run an assembly pipeline followed by an annotation step. Now, they have access to the gene positions and the corresponding strand orientations on the genome. If they are interested to compare the annotation results with those of other assembly pipelines, Idiom 2, which is our proposed circular multi-assembly annotation visualization can become handy. By uploading the output of the different annotation steps which need to be parsed into a CSV format containing the name of the Gene, starting position, ending position, and the strand, they will be able to visually compare the extracted information for the target assembly tools of interest or only one of them.

Scenario 3:
Imagine a bioinformatician needs to examine the quality, and more specifically, the contiguity of an assembled mitogenome resulting from a single assembly pipeline. If they also have access to the reference genome, using our tool, they can upload their assembly and reference files in the standard fasta format, and use Idiom 3, which helps them visually compare the two genomes.

Milestones

The planned schedule and milestones for this project have been provided in the table below.

Deadline	Task to be Completed	Assignments	Estimated Time (Hours Per Person)	Status
September 28, 2022	Project Pitch	Armaghan and Cecilia	2	Complete
October 3, 2022	Preparing the Annotation Results of One Species Sample Assembly (Sea Otter)	Cecilia	1	Complete
October 7, 2022	Parsing the Sea Otter Annotation Results	Armaghan	1	Complete
October 12, 2022	Pre-Proposal Meeting	Armaghan and Cecilia	-	Complete
October 14, 2022	Running the MitoZ and GetOrganelle Assembly Tools	Armaghan and Cecilia	8	Complete
October 21, 2022	Completion of the Project Proposal	Armaghan and Cecilia	5	Complete
October 31, 2022	Completion of Different Assembly Data Generation and Having Installed the Needed Packages/Libraries	Armaghan and Cecilia	20	Complete
November 8, 2022	Having the Initial Version of Visualizations Ready (GGisy Plots and HiPlots)	Armaghan and Cecilia	20	Complete
November 15, 2022	Written Updates Based on the Improvements and Changes in the Project After Evaluation of the Progress	Armaghan and Cecilia	6	Complete
November 16, 2022	Peer Project Reviews	Armaghan and Cecilia	-	Not Started Yet

November 22, 2022	Finalize the Gene Annotation Plot	Armaghan and Cecilia	20	In Progress
November 23, 2022	Post-Update Meeting	Armaghan and Cecilia	-	Not Started Yet
November 30, 2022	Finalize the FrontEnd Code(HTML + CSS)	Cecilia	20	Not Started Yet
December 10, 2022	Finalize the BackEnd Code (Python + Flask)	Armaghan	20	Not Started Yet
December 10, 2022	Finalize the Visualizations, Implementations and Analysis	Armaghan and Cecilia	5	In Progress
December 14, 2022	Finalize the Project Presentation	Armaghan and Cecilia	7	Not Started Yet
December 16, 2022	Finish and Submit the Final Paper	Armaghan and Cecilia	8	Not Started Yet

Future Work

Regarding the progress we have had so far in the project data generation and visualization implementation, here, we will discuss the work we are planning to do as for the next steps. First, the design of the Multi-Layer Genome Annotation Plot will be specified and implemented. As mentioned above, this will be a part of our mitogenome visualization toolbox. By finishing the implementation of this module, we will show the corresponding information of the proposed novel mitogenome assembly pipeline and the other two, namely, MitoZ, and GetOrganelle.

Next, the front end of our web-based application will be implemented, such that the user can select one of the visualization modules to use, and, accordingly, upload their data with the correct format. Having the graphical user interface of the webpage ready, we will start developing the back-end module, such that it parses the user's uploaded data and selected visualization tool and shows the result by executing the corresponding plotting program.

References

[1] Mikhail Alexeyev et al. "The Maintenance of Mitochondrial DNA Integrity – Critical Analysis and Update". In: *Cold Spring Harbour Perspectives in Biology* 5.5 (2013), a012641.

[2] David Sims et al. "Sequencing depth and coverage: key considerations in genomic analyses". In: *nature review genetics* 15 (2014), pp. 121–132.

[3] Sehi L'Yi et al. "Gosling: A grammar-based toolkit for scalable and interactive genomics data visualization". In: *IEEE Transactions on Visualization and Computer Graphics* 28.1 (2021), pp. 140–150.

[4] Helga Thorvaldsdóttir, James T Robinson, and Jill P Mesirov. “Integrative Genomics Viewer (IGV): high-performance genomics data visualization and exploration”. In: *Briefings in bioinformatics* 14.2 (2013), pp. 178–192.

[5] Maria Nattestad et al. “Ribbon: intuitive visualization for complex genomic variation”. In: *Bioinformatics* 37.3 (2021), pp. 413–415.

[6] Alla Mikheenko et al. “Icarus: visualizer for de novo assembly evaluation”. In: *Bioinformatics* 32.21 (2016), pp. 3321– 3323.

[7] Alexey Gurevich et al. “QUAST: quality assessment tool for genome assemblies”. In: *Bioinformatics* 29.8 (2013), pp. 1072–1075.

[8] René L Warren. “Visualizing genome synteny with xmatchview”. In: *bioRxiv* (2018), p. 238220.

[9] Martin Krzywinski et al. “Circos: an information aesthetic for comparative genomics”. In: *Genome research* 19.9 (2009), pp. 1639–1645.

[10] Hongen Zhang, Paul Meltzer, and Sean Davis. “RCircos: an R package for Circos 2D track plots”. In: *BMC bioinformatics* 14.1 (2013), pp. 1–5.

[11] Tengfei Yin, Dianne Cook, and Michael Lawrence. “ggbio: an R package for extending the grammar of graphics for genomic data”. In: *Genome biology* 13.8 (2012), pp. 1–14.

[12] Facebook Research. *HiPlot: High-dimensional interactive plots made easy*. https://ai.facebook.com/blog/hiplot-high-dimensional-interactive-plots-made-easy/. Accessed: 2022-10-18.

[13] Guanliang Meng et al. “MitoZ: a toolkit for animal mitochondrial genome assembly, annotation and visualization”. In: *Nucleic Acids Research* 47.11 (2019), e63.

[14] Moshi. *pyGenomeViz*. https://github.com/moshi4/pyGenomeViz. 2022.

[15] Jian-Jun Jin et al. “GetOrganelle: a fast and versatile toolkit for accurate de novo assembly of organelle genomes”. In: *Genome biology* 21.1 (2020), pp. 1–31.

[16] John D Hunter. “Matplotlib: A 2D graphics environment”. In: *Computing in science & engineering* 9.03 (2007), pp. 90–95.

[17] Michael L Waskom. “Seaborn: statistical data visualization”. In: *Journal of Open Source Software* 6.60 (2021), p. 3021.

[18] Sandro Valenzuela. *GGisy*. https://github.com/Sanrrone/GGisy. 2017.

[19] Daniel Bullock Hideto Mori Max Base. *pyCircos*. https://github.com/ponnhide/pyCircos. 2022.

[20] Oliver Schwengers Trestan Pillonel. *Mummer2Circos*. https:// github. com / metagenlab / mummer2circos. 2022.

[21] Grinberg, Miguel. *Flask web development: developing web applications with python*. " O'Reilly Media, Inc.", 2022