Linearização de Dados Espaço-Temporais Implementação e Avaliação de Curvas de Preenchimento de Espaço Guiadas por Dados

Gabriel de A. S. Meireles, Nivan Ferreira

Centro de Informática
Universidade Federal de Pernambuco (UFPE) – Recife, PE – Brazil

{gasm, nivan}@cin.ufpe.br

Abstract. *Static visualization of spatio-temporal data via linearization often relies on fixed-geometry curves, neglecting relevant data characteristics. This work presents an efficient, open-source library that implements and extends the Data-Driven Space-Filling Curves method to the temporal domain, introducing alignment techniques to ensure visual continuity between frames. Experiments indicate the effectiveness of the alignment in single-object scenarios, despite continuity limitations in simultaneous multi-object contexts. Finally, autocorrelation metrics validate the spatial and data coherence trade-offs described in the literature.*

Resumo. *A visualização estática de dados espaço-temporais via linearização frequentemente utiliza curvas de geometria fixa, ignorando características relevantes dos dados. Este trabalho apresenta uma biblioteca eficiente e de código aberto que implementa e estende o método de curvas de preenchimento guiadas por dados (Data-Driven Space Filling Curves) para o domínio temporal, introduzindo técnicas de alinhamento para garantir a continuidade visual entre quadros. Experimentos indicam a eficácia do alinhamento em cenários com um objeto único, apesar de limitações de continuidade em contextos com múltiplos objetos simultâneos. Por fim, métricas de autocorrelação validam os compromissos de coerência espacial e de dados previstos na literatura.*

1. Introdução

Os dados espaço-temporais, como o clima de uma região ou um padrão de propagação, podem ser visualizados de diversas formas. A abordagem mais comum para analisar um dado que varia com o tempo é o formato de vídeo ou uma sequência de quadros justapostos. Embora dados dispostos dessa maneira sejam de fácil visualização inicial, eles apresentam limitações para tarefas de análise aprofundada. Dentre os pontos negativos, destacam-se a dificuldade de perceber objetos que aparecem temporariamente e a necessidade do usuário manter o contexto mental para compreender a evolução temporal dos dados.

Recentemente, surgiram abordagens que utilizam linearizações do espaço para representar estaticamente dados temporais [2, 5], em especial com curvas de preenchimento de espaço (*space-filling curves*). A representação estática se dá por uma imagem bidimensional onde o eixo vertical representa a linearização de todo o espaço, enquanto o eixo horizontal representa o tempo. Essa representação é útil em casos onde animações

não são viáveis, como em impressões estáticas. Contudo, as abordagens presentes na literatura exploram dados espaço-temporais com curvas guiadas por dados [1, 9]. Em particular, a abordagem *data-driven* [9] geram curvas que possuem a propriedade de balancear a coerência dos dados com a coerência espacial, mantendo *features* próximas em sua linearização, o que pode ser uma estratégia interessante para evitar descontinuidades visuais.

Nesse contexto, este trabalho traz uma implementação eficiente, de fácil reuso e adaptação, do método *Data-Driven Space-Filling Curve* para imagens estáticas, bem como uma expansão do seu uso para dados espaço-temporais. O objetivo é gerar imagens estáticas representativas do evento temporal como um todo. Com o intuito de melhorar a coerência visual da imagem resultante, foram propostas também formas de alinhamento temporal entre os quadros.

A avaliação da proposta ocorre por meio de experimentos visuais em animações sintéticas. Adicionalmente, este trabalho implementa métricas de autocorrelação para aferir a coerência espacial e dos dados na linearização resultante.

2. Trabalhos Relacionados

A visualização de dados espaço-temporais é um tópico amplamente explorado na literatura. Peña-Araya et al. [6] conduziram estudos comparativos entre diferentes técnicas, especificamente: animações, pequenos múltiplos (*small multiples*) e glifos. Os resultados indicam que, embora animações sejam eficazes para estimar direções, a visualização estática via pequenos múltiplos apresenta melhor desempenho geral para tarefas de análise e comparação. Isso corrobora a relevância de abordagens que permitem a visualização simultânea de múltiplos instantes temporais, tal qual a proposta de linearização do espaço em uma dimensão com o tempo na outra.

As curvas de preenchimento de espaço possuem fundamentos teóricos sólidos [7], sendo a curva proposta por David Hilbert [3] uma das mais proeminentes na Ciência da Computação. A maioria dessas curvas tradicionais é construída com base em padrões geométricos fixos, sendo agnósticas ao conteúdo dos dados. Exceções importantes a essa regra são as curvas definidas por Dafner et al. [1] e Zhou et al. [9], que propõem abordagens guiadas por dados, onde o caminho se adapta às características da imagem para preservar a coerência das informações. Enquanto Dafner et al. definem a construção de caminhos hamiltonianos focados na maximização da coerência dos dados, Zhou et al. expandem esse conceito ao introduzir mecanismos para balancear simultaneamente a coerência espacial.

Recentemente, o uso de linearizações espaciais para representar dados temporais foi investigado por Franke et al. [2] e por Köpp e Weinkauf [5]. Franke et al. utilizam projeções 1D baseadas em curvas clássicas, como a de Hilbert, que não consideram a distribuição dos dados na imagem. Por outro lado, Köpp e Weinkauf desenvolvem um método próprio baseado em topologia. Nota-se, portanto, que a aplicação de curvas guiadas por dados especificamente para o contexto de visualização de sequências temporais ainda é pouco explorada, motivando a abordagem deste trabalho.

3. Metodologia

3.1. Visão Geral

A abordagem proposta fundamenta-se na linearização de estruturas multidimensionais (2D ou 3D) em sequências unidimensionais (1D). Expandindo o conceito de curvas baseadas em contexto (*Context-based Space-Filling Curves*) [1], utiliza-se a técnica de curvas de preenchimento de espaço guiadas por dados (*Data-Driven Space-Filling Curves*) [9]. O núcleo do processamento estabelece um ciclo hamiltoniano que percorre integralmente a grade de pixels, com o objetivo de preservar simultaneamente a coerência espacial, referente a localidade geométrica do caminho, e a coerência dos dados, associada à localidade dos dados no caminho.

Para viabilizar a implementação e a análise experimental, desenvolveu-se uma arquitetura de software híbrida entre C++ e Python. A metodologia trata imagens estáticas no formato $H \times W$ (2D) ou $H \times W \times C$ (3D). Analogamente, vídeos são processados como uma sequência de imagens estáticas do tipo $F \times H \times W$ ou $F \times H \times W \times C$, onde os parâmetros são definidos como:

- F: quantidade de quadros do vídeo;
- H: altura da imagem (número de linhas);
- W: largura da imagem (número de colunas);
- C: número de canais de cor.

O processamento de um vídeo ocorre, inicialmente, pela linearização independente de cada quadro. No entanto, curvas geradas de forma isolada para quadros consecutivos podem apresentar variações topológicas abruptas, prejudicando a coerência temporal. Para mitigar esse problema, o sistema implementa técnicas de alinhamento temporal, visando maximizar a correlação estrutural entre as curvas de frames adjacentes.

Dessa forma, o trabalho atua em duas frentes principais:

- **Imagens:** Implementação modular e reprodutível do método *data-driven* descrito por Zhou et al. [9] para grades regulares, visando a validação dos resultados teóricos;
- **Vídeos:** Investigação e avaliação de técnicas de alinhamento entre quadros para garantir a coerência temporal após a linearização individual de cada quadro do vídeo.

3.2. O método de linearização *data-driven*

A compreensão do método *data-driven* requer a análise preliminar da abordagem (*Context-based*) proposta por Dafner et al. [1]. O conceito central fundamenta-se no mapeamento da imagem original para o seu grafo dual, realizado através da construção de pequenos circuitos, conforme ilustrado na Figura 1.

Após esse mapeamento, observa-se que uma árvore geradora no grafo dual equivale a um ciclo hamiltoniano no grafo original. Dessa forma, busca-se definir uma função de peso para as arestas do grafo dual que considere os valores dos pixels da imagem. A partir dessa ponderação, constrói-se uma Árvore Geradora Mínima (MST) no grafo dual para, subsequentemente, recuperar o ciclo hamiltoniano no grafo original. Nesta implementação, adotou-se o algoritmo de Prim para a construção da MST.

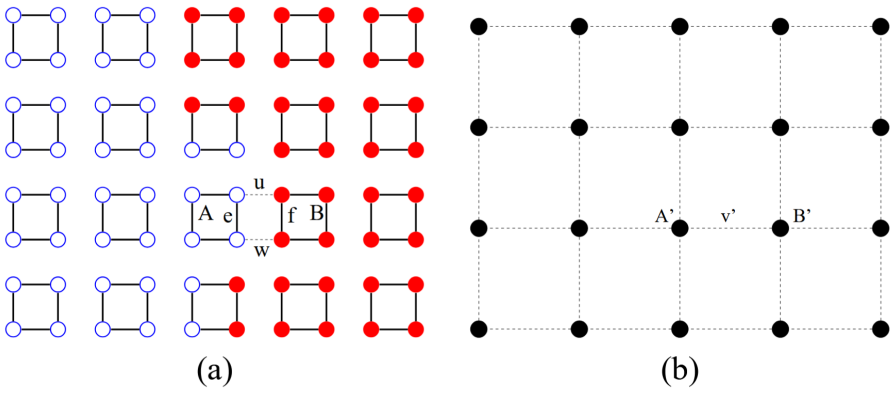


Figura 1. Pequenos circuitos de Dafner et al. [1]

A função de custo originalmente proposta é descrita na Equação 1. Nela, $W(C_i, C_j)$ representa o custo de conectar o vértice C_j à árvore geradora, partindo de um vértice C_i já pertencente a ela. As variáveis da equação (u, w, x, y, z, e, f) correspondem aos pesos das arestas no grafo original (ver Figura 2). O peso de cada aresta é definido pela diferença absoluta entre os valores dos pixels em seus extremos. Segundo Dafner et al. [1], para uma aresta $e = (a, b)$, tem-se $w(e) = |a - b|$. Para imagens coloridas (RGB), a diferença é a soma das diferenças absolutas de cada canal: $w(e) = \sum_{k \in \{R,G,B\}} |a_k - b_k|$.

Dessa maneira, a abordagem *context-based* gera linearizações que priorizam a coerência dos dados. Contudo, este método não considera explicitamente a coerência espacial durante a construção da curva. A abordagem *data-driven*, detalhada por Zhou et al. [9], soluciona essa limitação introduzindo uma nova função de custo composta, preservando, no entanto, a estrutura fundamental de mapeamento dual e MST proposta no artigo original [1].

A abordagem *data-driven*, conforme proposto em [9] utiliza a função de custo ponderada descrita na Equação 2. Os componentes da equação são definidos como:

- $N(C_i, C_j)$**:** Custo baseado na coerência dos dados, idêntico à função de Dafner et al. [1] (Equação 1);
- $R_\beta(C_j)$**:** Custo baseado na coerência espacial, definido pela distância euclidiana entre a célula C_j e o seu centróide S_{C_j}, *i.e.*, $||C_j - S_{C_j}||$;
- $\alpha \in [0, 1]$**:** Fator de ponderação. Valores menores de α priorizam a similaridade dos dados, enquanto valores maiores priorizam a regularidade espacial do caminho;
- β**:** Inteiro que define a dimensão $\beta \times \beta$ dos blocos no grafo dual, determinando a posição dos centroides S_{C_j} no centro de cada bloco. Experimentalmente, valores elevados de α tendem a confinar o ciclo hamiltoniano dentro de cada bloco antes de permitir a transição para regiões vizinhas.

Na implementação desenvolvida neste trabalho, α e β foram definidos como hiperparâmetros configuráveis, permitindo o ajuste fino do comportamento da curva e a análise do impacto que a variação desses parâmetros exerce sobre a preenchimento resultante. A relação geométrica e estrutural entre os custos de dados $N(C_i, C_j)$ e espacial $R_\beta(C_j)$ é ilustrada na Figura 3.

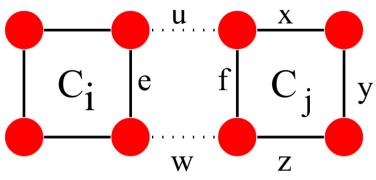


Figura 2. Pesos da função proposta de Dafner et al. [1]

$$W(C_i, C_j) = |u| + |w| + |x| + |y| + |z| - |e| - |f| \tag{1}$$

$$W(C_i, C_j) = (1 - \alpha)N(C_i, C_j) + \alpha R_\beta(C_j) \tag{2}$$

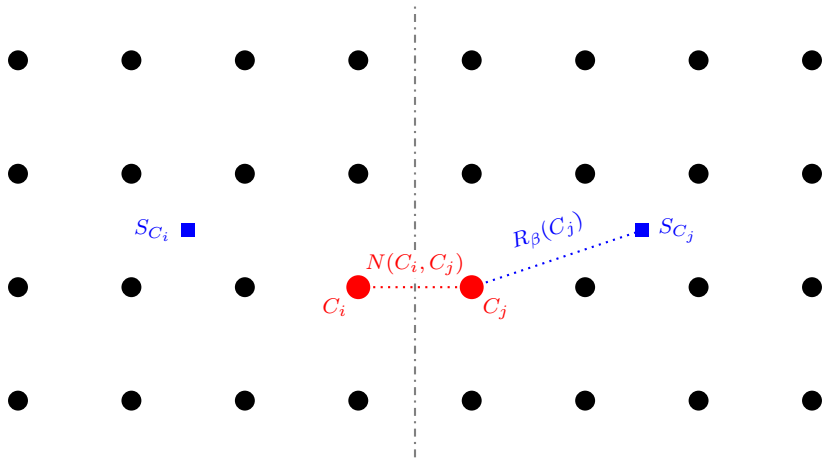


Figura 3. Representação das relações no grafo dual: o custo de dados $N(C_i, C_j)$ e o custo espacial $R_\beta(C_j)$.

3.3. Técnicas de Alinhamento Temporal

Entre dois quadros consecutivos, por mais semelhantes que sejam, pequenas variações nos valores dos pixels podem resultar em linearizações bem distintas. Esse fenômeno prejudica a análise da evolução temporal dos dados, pois quebra visualmente a continuidade entre os quadros. Para mitigar essa instabilidade e promover a coerência temporal na visualização, este trabalho explora duas técnicas de alinhamento. O método fundamenta-se na propriedade de que qualquer rotação cíclica ou inversão de sentido de um ciclo hamiltoniano constitui também uma representação válida de uma curva que cobre integralmente a imagem.

Dessa maneira, o problema de alinhamento consiste em encontrar, para o quadro atual, a rotação e a direção que minimizem uma função de custo objetiva em relação ao quadro anterior. Esse problema pode ser expresso como uma busca pelos parâmetros ótimos definidos na Equação 3. A função f presente nela é definida na Equação 4. Todos os parâmetros e seus significados são explicados a seguir:

- N: Tamanho do vetor da imagem linearizada;

- s: Índice do *cyclic shift* aplicado ao vetor;
- σ: Número binário que indica se o vetor foi invertido ou não;
- V_{t-1} e V_t: Vetores linearizados do quadro anterior e do atual;
- f: Função de custo global que quantifica a dissimilaridade total entre duas sequências;
- g: Função de custo local aplicada elemento a elemento.

$$(s^*, \sigma^*) = \underset{\substack{s\in[0,N-1]\\ \sigma\in\{0,1\}}}{\arg\min}\, f_\sigma(s) \tag{3}$$

$$f_\sigma(s) = \sum_{i=0}^{N-1} g\Big(V_{t-1}[i],\ V_{\sigma,t}[(i+s) \bmod N]\Big) \tag{4}$$

Neste trabalho, foram avaliadas duas métricas distintas para a função g: a *L1-norm* e a *L2-norm*. Para ambas, são funções facilmente extensíveis para imagens com múltiplos canais.

3.3.1. *L1-norm*

Nesta abordagem, a métrica local g é a diferença absoluta, conforme a Equação 5. Essa escolha mantém a consistência com a função de custo utilizada na construção da MST durante o algoritmo de Prim. Para imagens com múltiplos canais, a função é generalizada pelo somatório das diferenças em cada canal c, correspondendo à distância de Manhattan como destacado na Equação 6.

$$g(a, b) = |a - b| \tag{5}$$

$$g(a, b) = \sum_c |a_c - b_c| \tag{6}$$

Devido ao uso do módulo, a busca pela configuração ótima nesta estratégia ocorre de maneira exaustiva. Visto que para cada possível deslocamento é necessário computar a diferença em todos os pixels e canais, a complexidade resultante é $O(C \cdot N^2)$, onde C é o número de canais.

3.3.2. *L2-norm*

Nesta abordagem, a métrica local g é o quadrado da diferença, conforme a Equação 7. Similarmente, essa função pode ser generalizada para imagens com múltiplos canais como apresentado na Equação 8.

$$g(a, b) = (a - b)^2 \tag{7}$$

$$g(a,b) = \sum_{c} (a_c - b_c)^2 \tag{8}$$

Essa formulação permite expandir o termo quadrático da função global que se deseja minimizar e alcançar propriedades algébricas interessantes, como demonstrado na Equação 9:

$$\begin{aligned} f_\sigma(s) &= \sum_{i=0}^{N-1} \Big(V_{t-1}[i] - V_{\sigma,t}[(i+s) \bmod N] \Big)^2 \\ &= \sum_{i=0}^{N-1} V_{t-1}[i]^2 + \sum_{i=0}^{N-1} V_{\sigma,t}[(i+s) \bmod N]^2 - 2 \sum_{i=0}^{N-1} V_{t-1}[i] V_{\sigma,t}[(i+s) \bmod N] \\ &= \sum_{i=0}^{N-1} V_{t-1}[i]^2 + \sum_{i=0}^{N-1} V_t[i]^2 - 2 \underbrace{\sum_{i=0}^{N-1} V_{t-1}[i] V_{\sigma,t}[(i+s) \bmod N]}_{\text{correlação cruzada circular}} \end{aligned} \tag{9}$$

Dessa maneira, observa-se que, para alcançar o mínimo de f_σ, é preciso encontrar os valores de s e σ que maximizem a correlação cruzada circular [8], uma vez que os termos de soma quadrática são constantes e independem da rotação e da direção. Essa propriedade é fundamental pois, dado um $\sigma \in \{0, 1\}$, a correlação pode ser calculada para todo valor de s simultaneamente utilizando a Transformada Rápida de Fourier (FFT) em $O(N \log N)$. Como existem apenas duas orientações possíveis para σ, a complexidade assintótica se mantém. Assim, a complexidade final alcançada é $O(C \cdot N \log N)$, tal que C é o número de canais.

3.4. Detalhes de Implementação

Visando alto desempenho computacional, o núcleo de processamento numérico foi implementado inteiramente em C++20. Para garantir interoperabilidade e facilitar a análise exploratória, as funcionalidades foram expostas ao ambiente Python utilizando a biblioteca `Pybind11`, que atua como uma camada de integração leve.

O ambiente de validação experimental foi construído sobre *Jupyter Notebooks*, permitindo prototipagem rápida e visualização imediata dos resultados. Essa estrutura permitiu avaliar como a variação dos hiperparâmetros impacta o tempo de execução, a correlação entre quadros e a qualidade visual das curvas geradas.

O código-fonte desenvolvido encontra-se disponível publicamente no repositório: `https://github.com/gabmei/spatio-temporal-data-driven-sfc`.

4. Experimentos

4.1. Ambiente Computacional

Todos os testes foram realizados em um computador de uso pessoal, equipado com processador Intel Core i5-9300H 2.40 GHz, 16 GB de memória RAM e armazenamento em

SSD Samsung 980 PRO. A implementação foi executada em um ambiente virtual (venv) operando dentro do Subsistema Windows para Linux (WSL2).

É importante ressaltar que o ambiente WSL2 dispunha de apenas 8 GB de memória RAM e os experimentos não fizeram uso de técnicas de paralelismo. Essa restrição assegura que as métricas de tempo coletadas reflitam estritamente a complexidade algorítmica.

4.2. Conjuntos de Dados

Para a validação experimental, foi selecionado um conjunto diversificado de dados, provenientes de três fontes distintas. Ressalta-se que, para garantir a consistência nos experimentos, todos os dados foram normalizados para valores em ponto flutuante no intervalo $[0.0, 1.0]$.

4.2.1. Volumes do *Open Scientific Visualization Datasets*

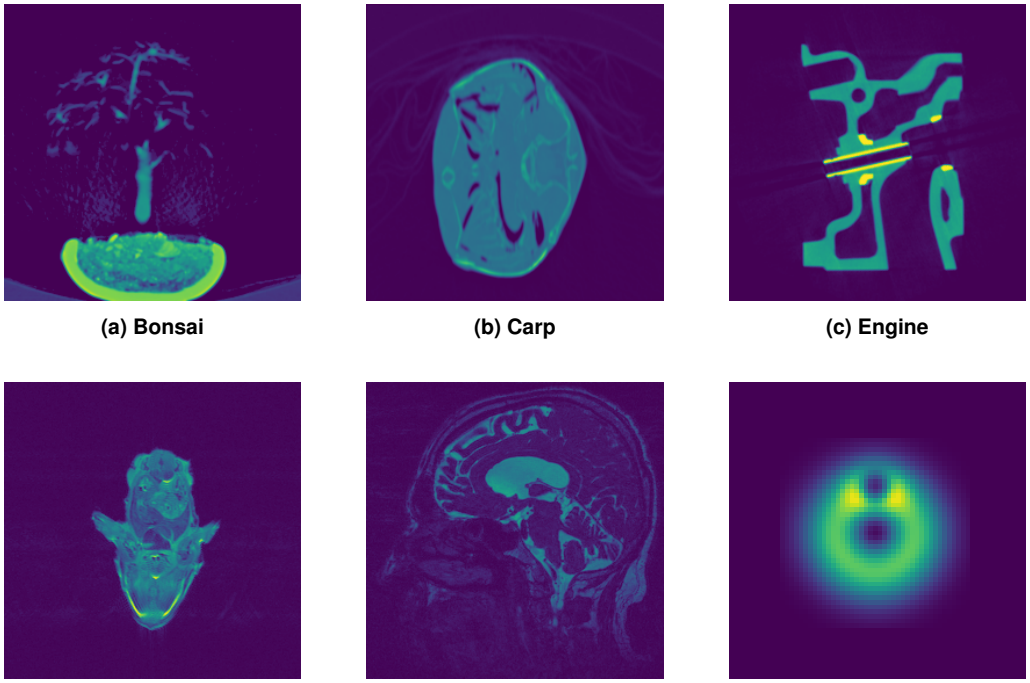
Para a validação experimental, foram utilizados diversos volumes de dados disponíveis na coleção *Open Scientific Visualization Datasets* [4], amplamente adotada na literatura, incluindo datasets usaods por Köpp e Weinkauf [5] e por Zhou et al. [9] em seus respectivos trabalhos. As especificações técnicas destes volumes encontram-se na Tabela 1. Cortes transversais ao longo do eixo z também são apresentados na Figura 4, destacando a diversidade dos dados.

Tabela 1. Conjuntos de dados do *Open SciVis*.

Dataset	Dimensões	Formato
Bonsai	$256 \times 256 \times 256$	uint8
Carp	$256 \times 256 \times 512$	uint16
Engine	$256 \times 256 \times 128$	uint8
Frog	$256 \times 256 \times 44$	uint8
Head MRI CISS	$256 \times 256 \times 124$	uint8
Nucleon	$41 \times 41 \times 41$	uint8

4.2.2. Imagens de Teste

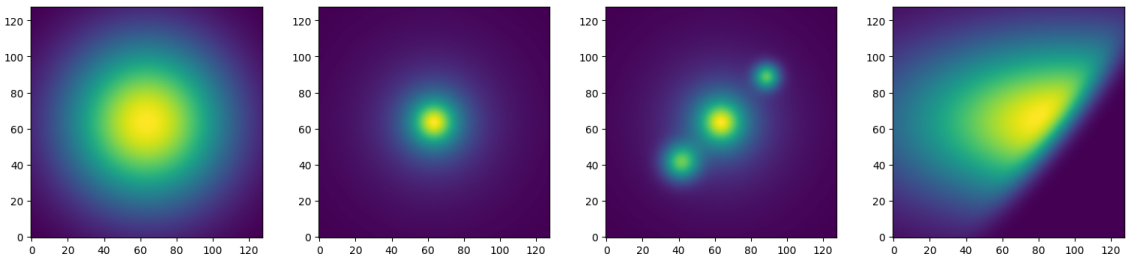
Para a validação do método em imagens estáticas, foram adotadas as figuras sintéticas propostas por Köpp e Weinkauf [5]. Estes dados possuem dimensões 128×128 com valores em ponto flutuante. Neste trabalho, tais imagens foram utilizadas para testar como diferentes distribuições de dados influenciam o comportamento da curva e como a variação de hiperparâmetros afeta o padrão de preenchimento resultante. A Figura 5 ilustra esses padrões.



(a) Bonsai **(b) Carp** **(c) Engine**

(d) Frog **(e) Head MRI** **(f) Nucleon**

Figura 4. Cortes transversais dos datasets utilizados nos experimentos, obtidos do repositório Open SciVis [4].



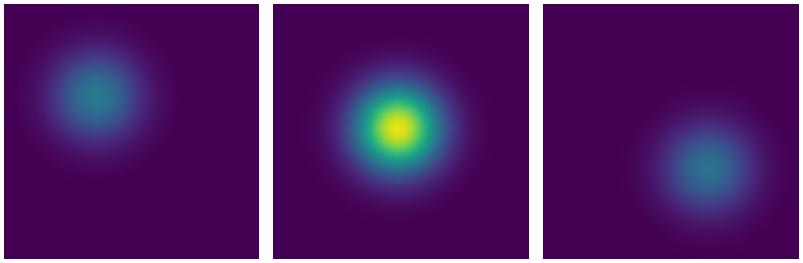
(a) Normal distribution **(b)** Delta sequence **(c)** Three features **(d)** Skew normal distribution

Figura 5. Imagens de teste de Köpp e Weinkauf [5]

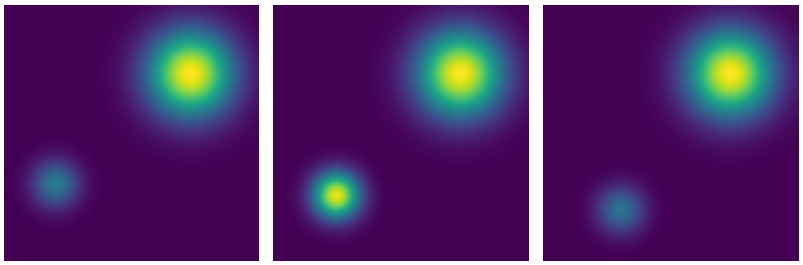
4.2.3. Animações Geradas

Para os experimentos espaço-temporais, foram gerados conjuntos de dados sintéticos especificamente para este trabalho, representando animações de objetos em diferentes cenários. Cada animação é composta por 60 quadros com resolução de 128×128 pixels, possuindo dimensão total $128 \times 128 \times 60$. A Figura 6 apresenta quadros selecionados destas animações. Os cenários são descritos a seguir:

- **Distribuição normal:** Uma distribuição normal em movimento diagonal, cuja intensidade surge e desaparece gradativamente;
- **Duas distribuições:** Duas distribuições normais, sendo uma estática de maior porte e outra menor em movimento diagonal, também com variação gradativa de intensidade;
- **Anel em expansão:** Um anel que cresce gradativamente em raio e intensidade. Este experimento replica o cenário proposto por Franke et al. [2] e utilizado por Köpp e Weinkauf [5].



(a) Distribuição normal



(b) Duas distribuições

(c) Anel em expansão

Figura 6. Evolução temporal das animações sintéticas geradas.

4.3. Avaliação de Linearização

Inicialmente, avalia-se a abordagem proposta em comparação com curvas de preenchimento estabelecidas na literatura, especificamente *Hilbert*, *Scanline* e *Context-based*.

Para este comparativo, foi adotada a imagem *Nucleon* da Figura 4(f), um dataset de referência frequentemente utilizado para validação de linearizações em trabalhos correlatos, como os de Zhou et al. [9] e Köpp e Weinkauf [5].

Em seguida, analisa-se a sensibilidade da implementação *data-driven* à variação de seus hiperparâmetros. Esse estudo é conduzido sobre os padrões sintéticos apresentados na Figura 5, o que permite a inspeção visual das alterações no padrão de preenchimento resultantes de diferentes configurações.

Por fim, replicam-se os testes estabelecidos por Zhou et al. [9], calculando-se a média da autocorrelação normalizada de duas métricas para analisar o desempenho das curvas. A normalização dos dados assegura que a correlação varie no intervalo $[-1, +1]$, o que possibilita a comparação justa entre diferentes datasets.

A métrica de média definida pelos autores consiste em um gráfico onde o eixo x representa o *lag* e o eixo y exibe a média dos valores de correlação computados na janela de $-lag$ a $+lag$. As métricas adotadas são:

- **Média Autocorrelação de Dados:** Considera a sequência das intensidades da linearização, *i.e*, $s(i) = image(P(i).x, P(i).y)$;
- **Média Autocorrelação Radial:** Considera a norma da posição espacial na linearização, *i.e*, $s(i) = ||(P(i).x, P(i).y)||$

4.3.1. Comparativo Visual no Dataset Nucleon

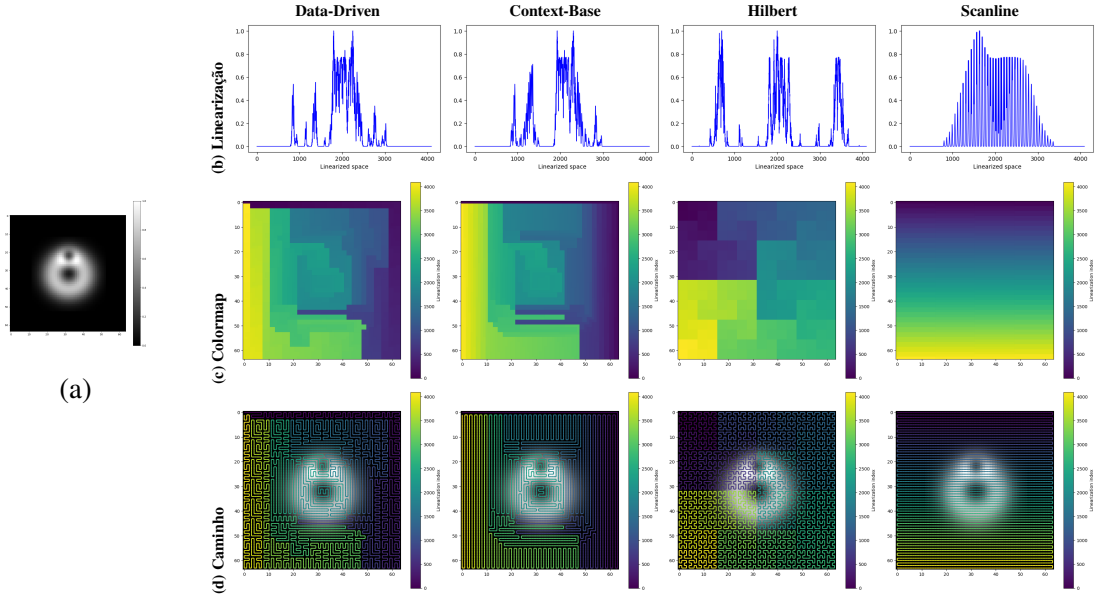


Figura 7. Comparação visual do método de linearização *data-driven* com demais métodos sobre o corte transversal do dataset Nucleon (a) inspirado por comparação similar em Zhou et al [9].

A Figura 7 apresenta a análise comparativa entre o método *data-driven* e as abordagens *context-based*, *hilbert* e *scanline* implementados neste trabalho. A avaliação foi conduzida sobre um corte transversal do volume *Nucleon* destacado na Figura 7a, que consiste de uma única feature com dois máximos locais próximos.

Uma representação da linearização resultante é exibida na linha **(b)**. Observa-se que tanto a abordagem *data-driven* quanto a *context-based* conseguem capturar a semântica dos dados, traduzindo os dois picos originais em elevações contínuas e distintas no gráfico. Em contrapartida, as curvas *hilbert* e *scanline*, por serem agnósticas aos dados, fragmentam essas estruturas, gerando um sinal oscilatório que não reflete diretamente a estrutura do dataset.

A distinção estrutural entre as abordagens guiadas por dados torna-se evidente na visualização do caminho percorrido, apresentada na linha **(d)**:

- Na abordagem ***context-based***, o algoritmo minimiza estritamente a diferença de valores destacados na Equação 1, o que frequentemente resulta em arestas longas e retilíneas que conectam pixels de intensidade similar.
- Na implementação ***data-driven***, a introdução do custo de coerência espacial presente na Equação 2 força a curva a ser mais compacta e sinuosa, preenchendo localmente uma região antes de transitar para a próxima.

Este comportamento sinuoso da abordagem *data-driven* e as linhas retilíneas da abordagem *context-based* corroboram os resultados apresentados originalmente por Zhou et al. [9]. Vale ressaltar que a formação desses padrões geométricos repetitivos decorre da natureza determinística do algoritmo de Prim. Em situações em que múltiplos vizinhos apresentam custo idêntico na Equação 2, a ordem de precedência das direções (norte, sul, leste, oeste) atua como critério de desempate, induzindo a curva em direções preferenciais.

4.3.2. Análise de Sensibilidade a Hiperparâmetros

A Figura 8 destaca o impacto de alterações nos hiperparâmetros da Equação 2 do método *data-driven*. Diferentemente das limitações discutidas por Köpp e Weinkauf [5], nota-se que a abordagem *data-driven* gera linearizações bem distintas entre si, o que indica sua capacidade de diferenciar tais distribuições. É perceptível também que o aumento do hiperparâmetro α induz uma maior segmentação na linearização resultante, pois força a curva a percorrer a imagem em blocos. Em especial, nota-se que a capacidade de distinguir o número de estruturas em 8(c) é gradativamente atenuada ao se aumentarem os valores dos hiperparâmetros, visto que a combinação de α e β pode fragmentar uma mesma estrutura em múltiplos segmentos do caminho.

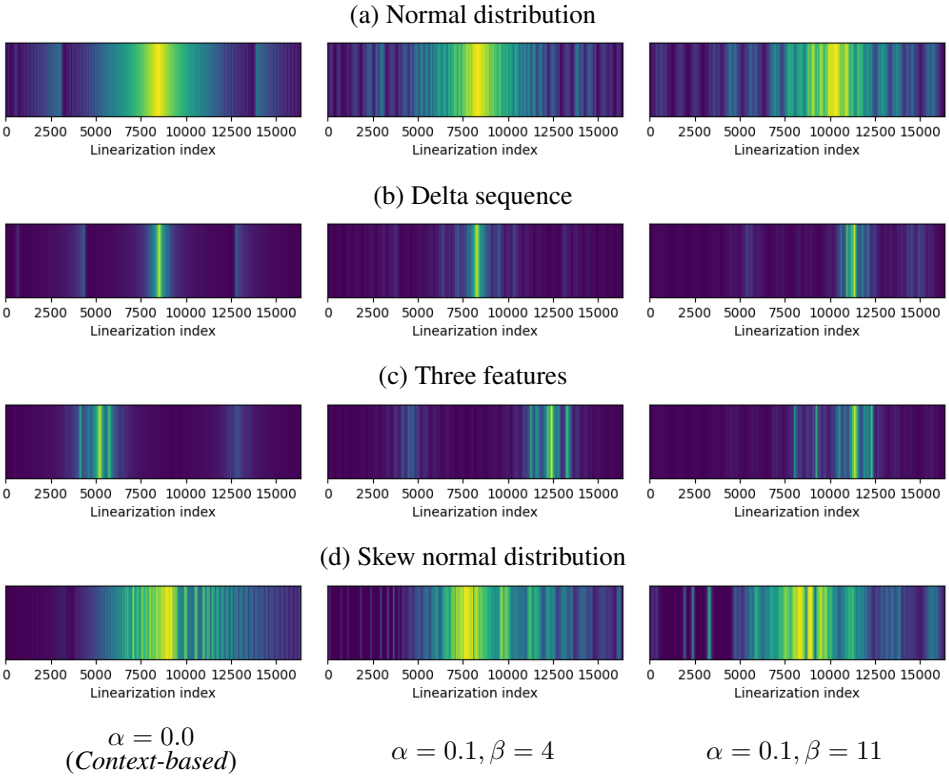
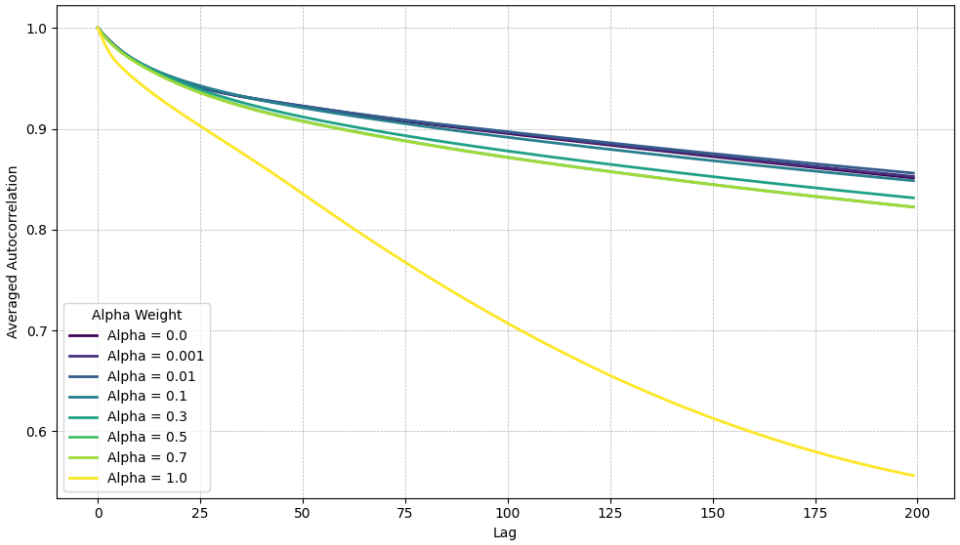


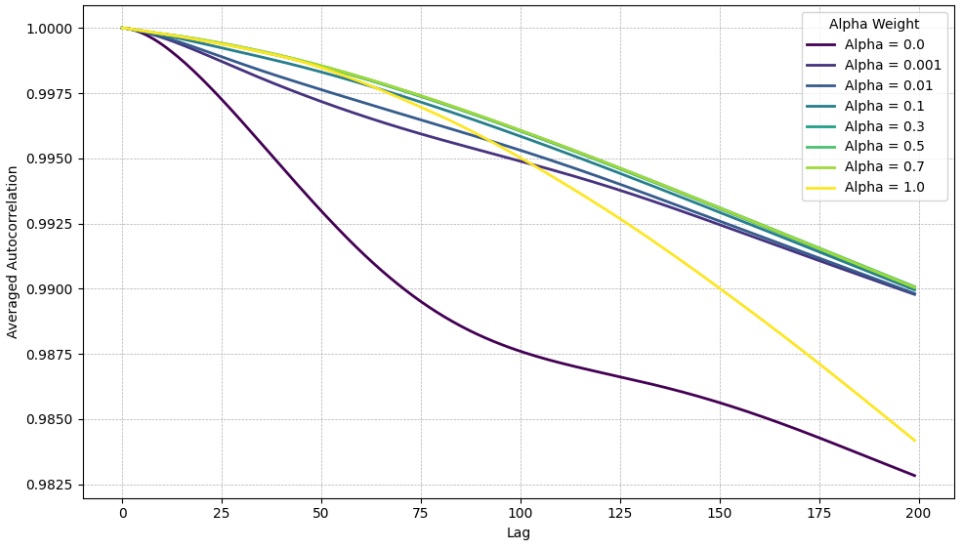
Figura 8. Análise de sensibilidade aos hiperparâmetros sobre os padrões sintéticos de Köpp e Weinkauf [5] presentes na Figura 5. As colunas evidenciam o impacto das variações de α e β na linearização resultante.

4.3.3. Avaliação Quantitativa via Autocorrelação

Para a avaliação quantitativa da autocorrelação, estabeleceu-se um conjunto de dados composto por 10 cortes transversais de cada volume do *Open SciVis* presentes na Tabela 1, somados a 10 quadros das animações sintéticas apresentadas na Figura 6. Nos experimentos subsequentes, manteve-se o valor de β fixo, variando-se exclusivamente o hiperparâmetro α. Para cada imagem selecionada, foram computadas as métricas de autocorrelação de dados e radial, obtendo-se a média global dos resultados para as diferentes configurações avaliadas.

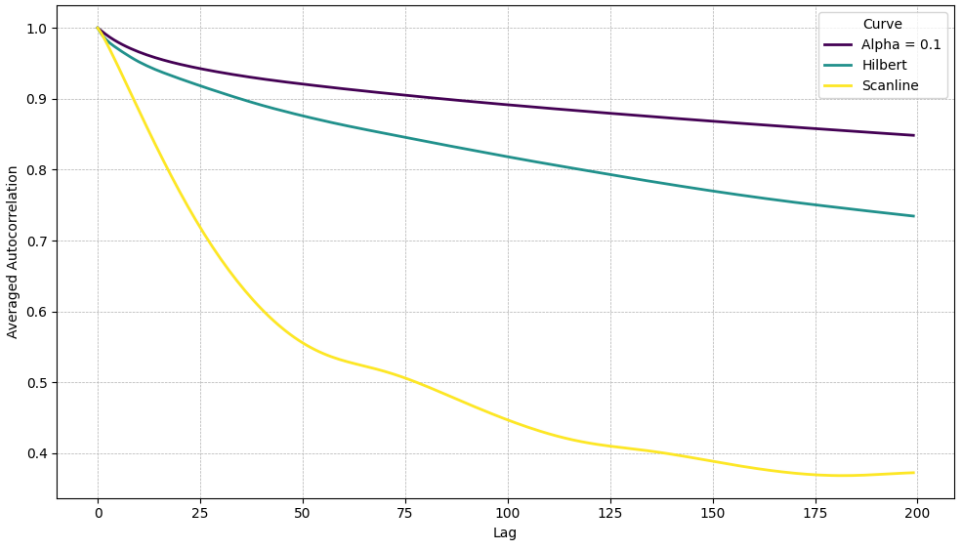


(a) Média de Autocorrelação de Dados

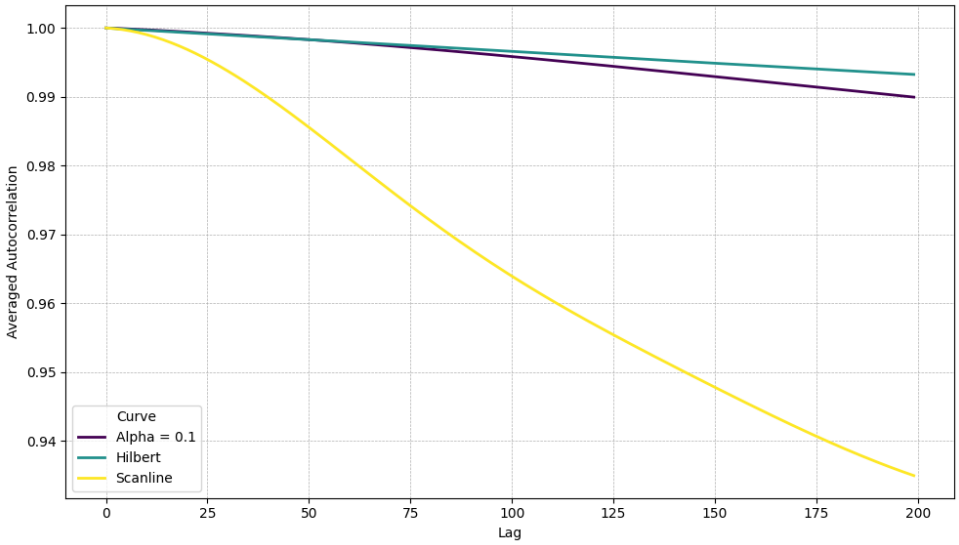


(b) Média de Autocorrelação Radial

Figura 9. Impacto da variação de α na autocorrelação média normalizada. (a) A métrica de dados avalia a continuidade das intensidades. (b) A métrica radial avalia a preservação da vizinhança espacial.



(a) Média de Autocorrelação de Dados



(b) Média de Autocorrelação Radial

Figura 10. Comparação da autocorrelação média normalizada entre as abordagens *data-driven*, *hilbert* e *scanline*. (a) A métrica de dados avalia a continuidade das intensidades. (b) A métrica radial avalia a preservação da vizinhança espacial.

Os resultados comparativos dessa avaliação são detalhados nas Figuras 9 e 10.

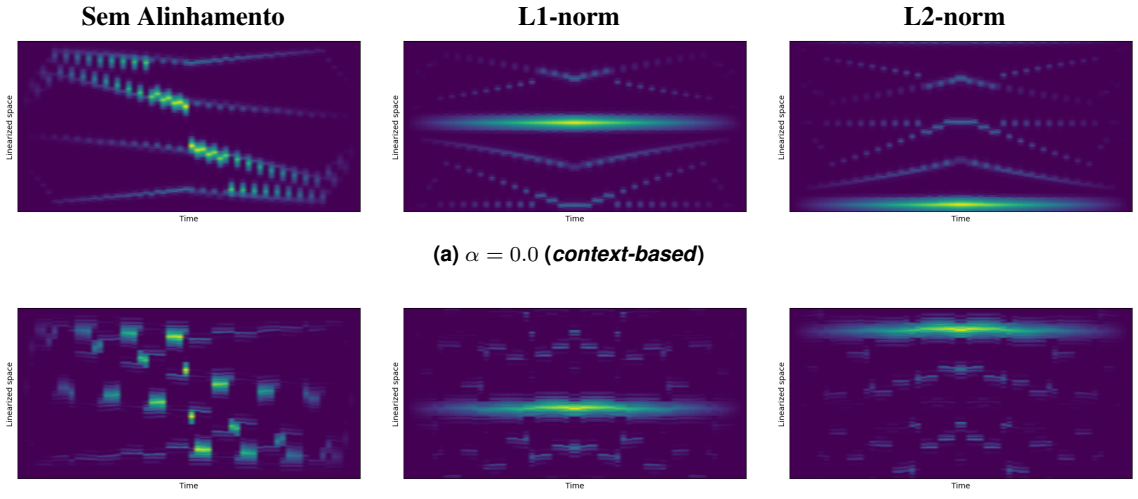
A Figura 9 ilustra o comportamento médio das curvas em relação ao deslocamento (*lag*), evidenciando o *trade-off* imposto pelo hiperparâmetro α, cujo incremento reduz a autocorrelação de dados ao passo que amplia a autocorrelação radial. Observa-se, contudo, uma degradação acentuada nas métricas para o caso limite de $\alpha = 1$. Isso ocorre pois, neste extremo, o método negligencia o termo dos dados em favor do termo puramente espacial da Equação 2, o que aproxima o comportamento da curva ao do algoritmo *scanline* ao percorrer a imagem em blocos rígidos.

Por sua vez, a Figura 10 confronta a abordagem *data-driven* para o valor fixo $\alpha = 0.1$ com as curvas de hilbert e *scanline*. Os resultados corroboram as observações de Zhou et al. [9], pois nota-se que o ganho na coerência espacial aproxima o método proposto do desempenho da curva de hilbert, ainda que isso acarrete uma leve penalidade na autocorrelação de dados.

4.4. Avaliação de Animação

O resultado do processamento espaço-temporal constitui uma imagem 2D, onde o eixo vertical corresponde ao espaço linearizado de cada quadro e o eixo horizontal representa a evolução temporal. A resolução final é dada por $m \times n$, sendo m o número total de pixels por quadro e n a quantidade quadros.

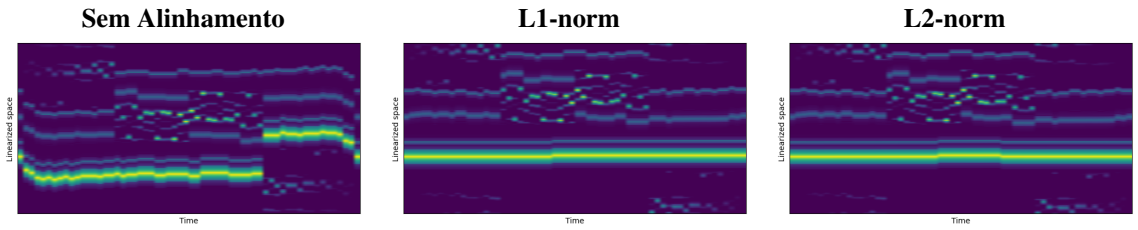
A eficácia dos alinhamentos temporais para as animações da Figura 6 foi validada visualmente. A análise comparativa confronta os resultados obtidos para $\alpha = 0.0$ e $\alpha = 0.1$ sob diferentes estratégias de alinhamento (Sem alinhamento, Normas L1 e L2), mantendo-se fixo o hiperparâmetro β.



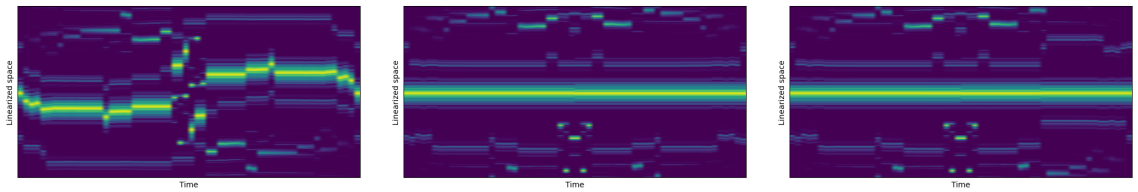
(a) $\alpha = 0.0$ (***context-based***)

(b) $\alpha = 0.1$ (***data-driven***)

Figura 11. Linearizações para a animação da Distribuição Normal visualizada na Figura 6a.

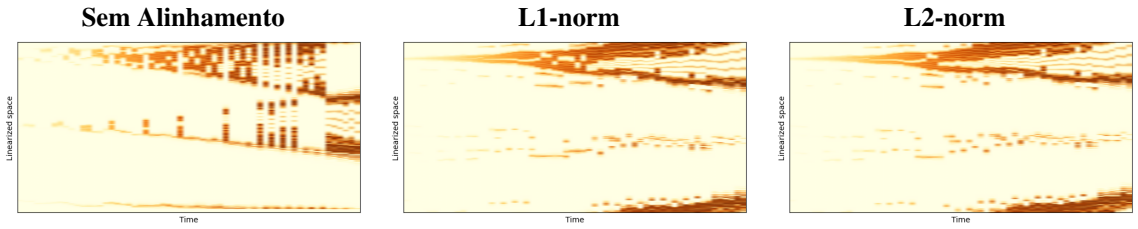


(a) $\alpha = 0.0$ (***context-based***)

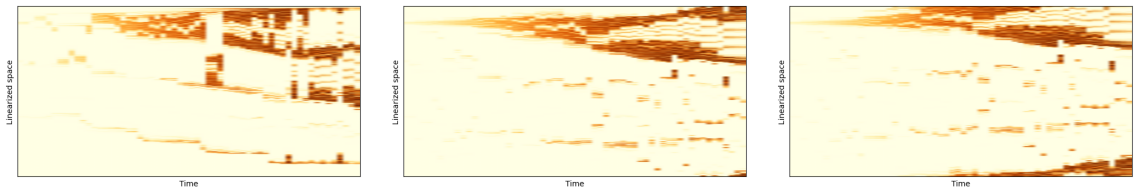


(b) $\alpha = 0.1$ (***data-driven***)

Figura 12. Linearizações para a animação de Duas Distribuições visualizada na Figura 6b.



(a) $\alpha = 0.0$ (***context-based***)



(b) $\alpha = 0.1$ (***data-driven***)

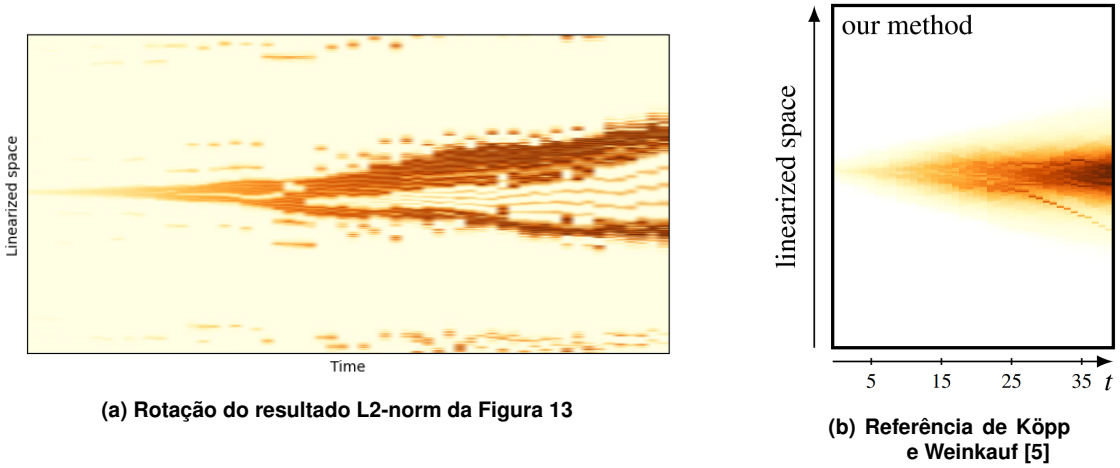
Figura 13. Linearizações para a animação do Anel em Expansão visualizada na Figura 6c.

As Figuras 11, 12 e 13 apresentam as imagens resultantes para cada configuração avaliada. É perceptível que, independentemente da técnica aplicada, a abordagem *data-driven* tende a fragmentar as *features* presentes nos dados, efeito que se intensifica com o aumento do termo de regularização espacial α.

Embora a abordagem *data-driven* preserve a conectividade das *features* dentro de um único quadro, a instabilidade da representação entre quadros consecutivos dificulta o alinhamento temporal em cenários com múltiplos objetos. Isso é evidenciado na Figura 12, onde a fragmentação excessiva torna impossível inferir visualmente a quantidade de *features* existentes na animação.

Em contrapartida, as Figuras 11 e 13 são cenários que contêm apenas uma estrutura de interesse. Nestes casos, as imagens espaço-temporais geradas preservam a coerência da animação original, mesmo sob maior influência do fator espacial α da Equação 2.

Além disso, é importante notar que cada quadro da animação representa um circuito hamiltoniano fechado. Dessa forma, aplicar uma mesma rotação cíclica nos quadros preserva a coerência do alinhamento resultante da mesma maneira. Essa propriedade é demonstrada na Figura 14, onde uma versão rotacionada do resultado obtido é comparada diretamente com o padrão da literatura reportado por Köpp e Weinkauf [5].



(a) Rotação do resultado L2-norm da Figura 13

(b) Referência de Köpp e Weinkauf [5]

Figura 14. Comparação visual entre o resultado rotacionado do método proposto (a) e o resultado reportado na literatura (b) para o cenário do Anel em Expansão.

4.5. Tempos de execução

Para avaliar o desempenho computacional, calculou-se a média dos tempos de execução após 20 repetições para cada experimento. Nessas medições, os conjuntos de imagens de teste foram gerados de maneira aleatória. O primeiro experimento, apresentado na Figura 15, avalia os custos temporais associados à geração do caminho linearizado para uma única imagem. O gráfico relaciona o número de pixels da imagem no eixo horizontal com o tempo de execução em milissegundos no eixo vertical. As legendas são detalhadas a seguir:

- ***C++ core algorithm*:** Custo computacional exclusivo da execução do algoritmo de Prim sobre a imagem;

- ***Pybind interface overhead*:** Custo de tempo para pré-processar a entrada e adequá-la ao formato da implementação;
- ***Python overhead*:** Custo adicional referente à mudança de contexto entre as linguagens de programação.

O que pode ser observado pela Figura 15 é que o principal custo de toda a execução está no processamento da imagem durante o algoritmo de Prim, o que indica que não existem perdas de performance significativas ao adotar a biblioteca Pybind para expor a implementação de C++ para Python.

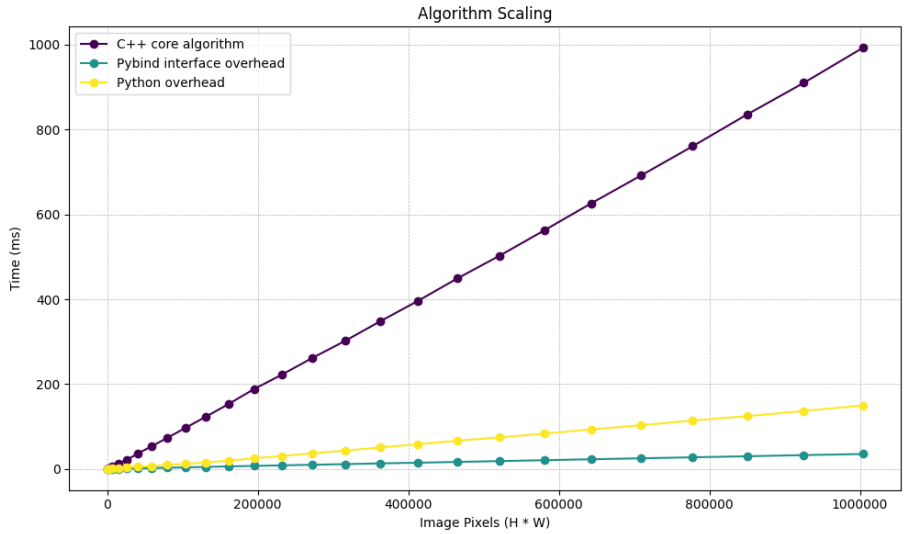


Figura 15. Tempo de execução para calcular o caminho hamiltoniano de uma imagem única, discriminando os custos entre C++, Pybind e Python

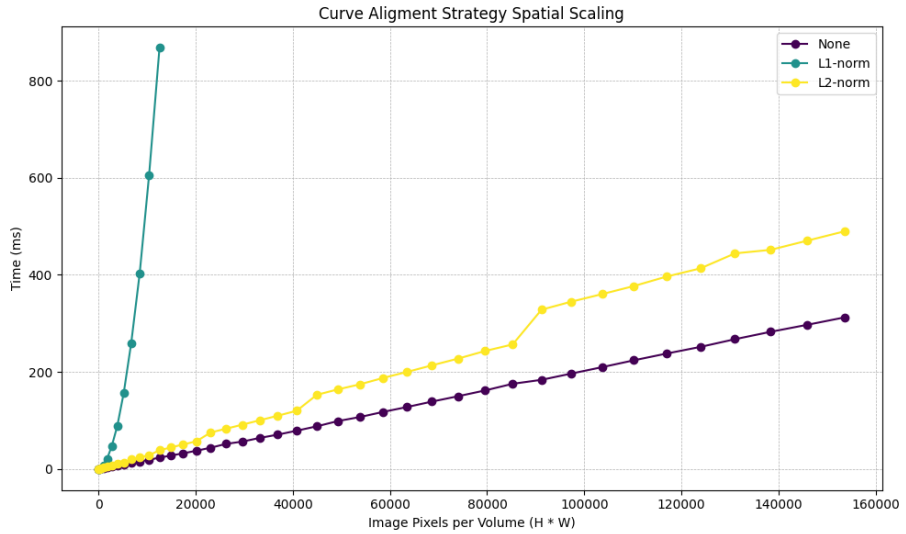


Figura 16. Impacto do aumento do número de pixels no tempo de execução para diferentes estratégias de alinhamento

As Figuras 16 e 17 avaliam o tempo de execução total decorrente da estratégia de alinhamento escolhida. Na Figura 16, considera-se um volume de dados com apenas

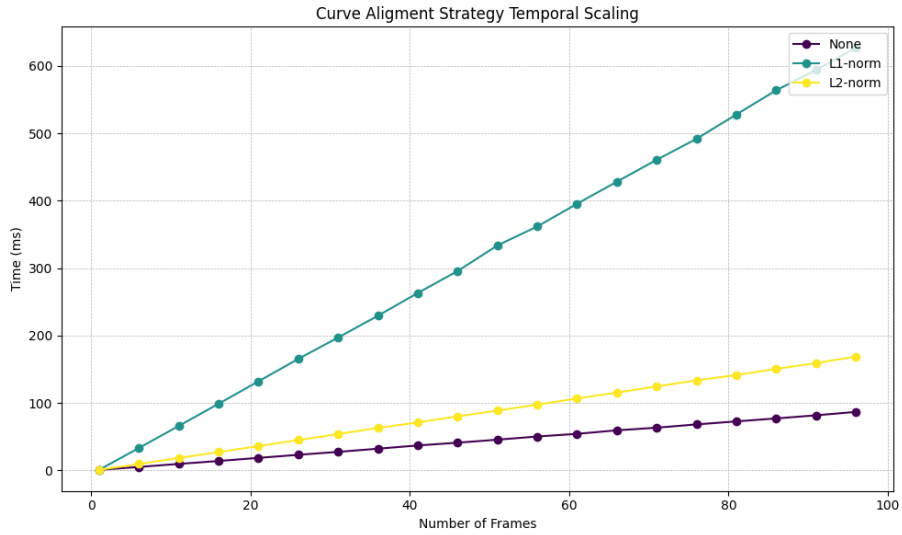


Figura 17. Impacto do aumento do número de quadros no tempo de execução para diferentes estratégias de alinhamento

dois quadros, variando-se o número de pixels em cada um deles. Observa-se o elevado custo da estratégia *L1-norm*, que exibe complexidade quadrática em relação ao número de pixels, enquanto a estratégia *L2-norm* introduz pouco custo adicional, pois possui a mesma ordem de complexidade do algoritmo de Prim.

Por sua vez, na Figura 17, mantém-se fixo o número de pixels por quadro em 32×32, variando-se o número total de quadros. Observa-se que as estratégias apresentam comportamento linear em relação à quantidade de quadros, uma vez que o processamento ocorre de maneira independente para cada quadro da animação.

5. Conclusão

Este trabalho apresentou o desenvolvimento e a análise de uma implementação eficiente do método de linearização *data-driven*, expandindo sua aplicação para o domínio espaço-temporal. A arquitetura híbrida adotada permitiu combinar a eficiência de C++ para a implementação dos algoritmos utilizados com a flexibilidade de Python para a validação de hiperparâmetros e métricas de qualidade.

Os experimentos confirmam o *trade-off* imposto pelo fator de regularização espacial α, onde valores maiores promovem curvas de preenchimento de espaço em blocos compactos, em detrimento da coerência dos dados.

No contexto das animações, foram introduzidas duas estratégias de alinhamento espaço-temporal entre quadros: *L1-norm* e *L2-norm*. Os resultados indicam a necessidade dessas técnicas, uma vez que sua ausência gera figuras desconexas. A adoção dessas estratégias mostrou-se efetiva em cenários simples, onde há apenas uma *feature* de interesse ao longo da animação. Contudo, observou-se uma limitação em cenários contendo múltiplos objetos simultâneos. Mesmo que o método *data-driven* seja capaz de linearizar adequadamente cada quadro individualmente, a instabilidade na ordenação das *features* entre quadros consecutivos prejudica o alinhamento temporal.

Adicionalmente, notou-se que a implementação de referência de Zhou et al. [9] e

a proposta neste trabalho adotam direções preferenciais determinísticas para o desempate da Equação 2 durante a construção da MST. Essa característica pode introduzir vieses direcionais sutis na linearização resultante.

Como direções para trabalhos futuros, sugere-se:

- Analisar como estratégias de desempate da Equação 2, como a aleatória, afetam as métricas de autocorrelação;
- Adaptar as métricas de autocorrelação para avaliar formalmente a consistência temporal;
- Explorar novas técnicas de alinhamento, como considerar uma janela temporal em vez de apenas o quadro anterior e a inclusão de um fator espacial durante o alinhamento.

Por fim, o código-fonte está disponível e é facilmente customizável, oferecendo à comunidade uma ferramenta para exploração de técnicas de linearização e visualização de dados espaço-temporais.

Referências

[1] Revital Dafner, Daniel Cohen-Or e Yossi Matias. "Context-based space filling curves". Em: *Computer Graphics Forum* 19.3 (2000), pp. 209–218. DOI: 10.1111/1467-8659.00413.

[2] Max Franke et al. "Visual Analysis of Spatio-temporal Phenomena with 1D Projections". Em: *Computer Graphics Forum* 40.3 (jun. de 2021), pp. 335–347. ISSN: 1467-8659. DOI: 10.1111/cgf.14311.

[3] David Hilbert. "Ueber die stetige Abbildung einer Line auf ein Flächenstück". Em: *Mathematische Annalen* 38.3 (1891), pp. 459–460. DOI: 10.1007/BF01199431.

[4] Pavol Klacansky. *Open SciVis Datasets*. http://klacansky.com/open-scivis-datasets/. Acessado em: 8 de janeiro de 2026. 2017.

[5] Wiebke Köpp e Tino Weinkauf. "Temporal Merge Tree Maps: A Topology-Based Static Visualization for Temporal Scalar Data". Em: *IEEE Transactions on Visualization and Computer Graphics (Proc. IEEE VIS)* 29.1 (jan. de 2022). DOI: 10.1109/TVCG.2022.3209387.

[6] Vanessa Peña-Araya, Anastasia Bezerianos e Emmanuel Pietriga. "A Comparison of Geographical Propagation Visualizations". Em: *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. CHI '20. ACM, 2020, pp. 1–14. DOI: 10.1145/3313831.3376350.

[7] Hans Sagan. *Space-Filling Curves*. New York: Springer-Verlag, 1994. DOI: 10.1007/978-1-4612-0871-6.

[8] Chen Wang. "Kernel learning for visual perception". Capítulo 2.2.1, pp. 17–18. Tese de dout. Singapore: School of Electrical e Electronic Engineering, Nanyang Technological University, 2019. DOI: 10.32657/10220/47835. URL: https://hdl.handle.net/10220/47835.

[9] Liang Zhou, Chris R. Johnson e Daniel Weiskopf. "Data-Driven Space-Filling Curves". Em: *IEEE Transactions on Visualization and Computer Graphics* 27.2 (2020), pp. 1591–1600. DOI: 10.1109/TVCG.2020.3030473.