VLMを用いたタイムシートスキャン画像のOCR

前島 謙宣[1,a)]　品川 政太朗[2,b)]　舩冨 卓哉[3,2,c)]　向川 康博[2,d)]

概要

アニメ制作において, タイムシートはプロダクションの各工程に必要な情報を伝達する上で重要な役割を果たしている．制作のデジタル化が推し進められてはいるものの，自由度の高さから紙のシートも併用されているのが現状である．デジタルコンテンツクリエーションツール上に指定されたタイミングを再現する際に，手作業での入力が求められ，これが作業者にとっての負担となっている．本論文では，紙のタイムシートのスキャン画像に対する OCR フレームワークを提案する．タイムシートの種類やマス目などの構造解析には決定論的な CV 技術を用い，解釈や曖昧性に対処しなければならない文字や記号の読み取りには，タイムシートの記法を指示した視覚言語モデルを用いる．実際の制作に使われた小規模なタイムシートのデータセット上で認識実験を行い，提案するフレームワークの有効性を示す．

1. はじめに

アニメ制作において，タイムシートは原画，動画，仕上げ，撮影といったプロダクションの各工程において必要な情報を伝達する重要な役割を果たしている．具体的には，原画と動画の枚数やセルの重ね方，セリフ，カメラワーク，特殊効果の指示などが記載されており，例えば，セルを合成する撮影工程の担当者は，このタイムシートを参照しながら，Adobe After Effects のような合成ソフトウェア上のタイムラインにセルを並べて合成作業を行う．

制作工程のデジタル化が進む昨今，東映アニメーションデジタルタイムシート [2] のように，タイムシートもデジタル化が進展している一方で，現場では自由度の高さから未だに紙のシートも併用されており，シートのスキャン画像がやりとりされていることも多い．画像ではなく，コンピュータで編集・検索可能なデジタルテキストに自動変換できる技術を確立することは，今の作業者の負担を軽減するだけでなく，作品の中間素材のデジタルアーカイビングの観点からも，重要であると言える．

タイムシートには，図 1 に示すような従来の OCR 技術では認識が困難な要素がいくつか存在する．例えば，文字コード変換できない三角形の囲み文字が書かれていたり，直前の原画あるいは動画が継続することを意味するマス目を縦断する直線や波線が書かれていたりする．また，手書きで，マス目を飛び出して書かれることも多い．最近では，視覚言語モデル（VLM）を用いた OCR も存在し，手書き文字についても高精度な認識を実現しているが，検出された文字や記号のバウンディングボックスを正確に取ることが難しく，タイムシートのような「どこに何が書かれていたか」を正確に把握する必要がある場合には，まだ実用において課題が残る．これらの課題に対処し，タイムシートの高精度な OCR を実現することは未だチャレンジングな問題であると言える．

本論文では，従来の CV 技術と最先端の VLM を組み合わせたタイムシートスキャン画像に対する自動認識フレームワークを提案する．提案するフレームワークでは，タイムシートの種類やマス目といった構造的な特定には決定論的な CV 技術を用い，意味を考慮したり曖昧性に対処しなければならない文字や記号の読み取りには確率的な VLM を用いる．実際の制作に使われた小規模なタイムシートのデータセット上で認識実験を行い，提案するフレームワークの有効性を示す．

本論文の貢献は以下の 3 点である．

- CV 技術と VLM の組み合わせによる高精度なタイムシートの自動認識フレームワークを提案した．
- タイムシートのマス目に沿った認識を促し，構造化出力を安定化させるための Visual Prompting 手法を提案した．
- 小規模なタイムシートのデータセット上で平均完全一致率 90.1%を達成した．

2. タイムシートの構成

図 1 に本研究で対象とするタイムシートの一例を示す．タイムシートは，どの作品のどのカットが何秒 (たいてい

[1] 株式会社オー・エル・エム・デジタル
[2] 奈良先端科学技術大学院大学
[3] 京都大学
[a)] akinobu.maejima@olm.co.jp
[b)] sei.shinagawa@is.naist.jp
[c)] funatomi.takuya.2c@kyoto-u.ac.jp
[d)] mukaigawa@is.naist.jp

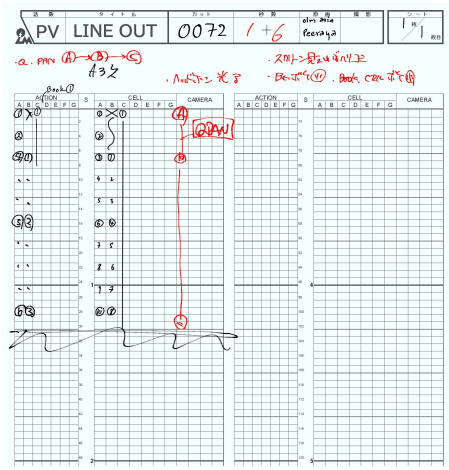

図 1 タイムシートの一例

の場合 1 秒 24 コマ）何コマの時間尺で表現されているか，また原画（ACTION）と動画（CELL）の各セル（この例の場合 A から G まで）を構成する画像がどのコマに割り当てられているかが，記号と数字およびカタカナによって表現される．ACTION および CELL に書かれる数字やカタカナは，原画や動画の画像番号（名前）に対応している．ACTION で，数字やカタカナが丸で囲まれている場合は，その画像が原画であることを意味し，CELL においても対応する数字が丸で囲まれる．ACTION の・は，動画においてそのコマを中割りする必要があることを意味しており，その結果生まれた画像は，CELL の対応する行で単なる数字として表される．空欄は直前のコマ（画像）が継続することを表し，その状態が 3 コマ以上継続する場合は縦の直線で記される．本論文ではこの線を**継続線**と呼ぶ．また，空セルと何も画像を表示しないという指示も ACTION, CELL 同じタイミングで記される．つまり ACTION と CELL の間には，記法における一定の関連性がある．

3. 関連研究

著者らの知る限り，現時点でアニメ制作におけるタイムシートの OCR を扱った研究は存在しない．タイムシートは各欄のどこに何が書かれているのかを，組として認識する必要があるため，文字や記号とそれらが書かれているバウンディングボックスをまとめて正確に認識する必要がある．Brandon らは，表の検出と行・列の識別が可能な Table Transformer[5] を提案している．タイムシートの表のフォーマットは，多少のバリエーションはあるもののおおよそ決まっている．このため本研究では何も書いていないシートとのテンプレートマッチングとホモグラフィ変換によりテンプレートの構造を入力にあてはめ，表の行・列を認識する．

TrOCR[3] などの Transformer ベースの手法の登場により，OCR も高精度が進んでいるが，タイムシートに書かれる原画番号や動画番号，あるいはそれ以外の記号については，直接マッピングすることが不可能な文字や記号がある．本研究では，OCR に VLM を用いる．特に原画番号で使われる，直接のマッピングが不可能な丸や三角形で囲まれた数字・カタカナ・記号の囲み文字の OCR を，数字とそれを囲む図形記号としてそれぞれ独立に VLM に認識させることで対処する．提案するフレームワークにおける VLM は，Few-shot Prompting と構造化出力がサポートされているモデルであれば任意のモデルで良い．

4. 提案手法

提案するフレームワークの概要を図 2 に示す．事前準備として，システムに入力される可能性のあるタイムシートのパターンを事前に収集し，タイムシートのテンプレートプールを構成しておく（4.1 節）．またそれぞれについて，ACTION 欄，CELL 欄のセル毎に縦方向（時系列方向）に並んだマスをグルーピングしておく．タイムシートのスキャン画像（RGB 画像）が入力されると，テンプレートとの照合が行われ，最も類似したテンプレートが選択される．このとき同時にテンプレートへの位置合わせが行われる（4.2 節）．次に，入力とテンプレートを比較し，認識すべき文字の候補と継続線を検出 (4.3 節) する．その後，ACTION 欄と CELL 欄の双方に対して重なりのない固定長のスライディングウィンドウを設け，ウィンドウの範囲内のマスの画像を抽出し，左右に結合する．このとき VLM がマスと元画像上のマスの位置を認識できるように Visual Prompting として画像の結合時にマス目や文字と異なる色でマス目の順番をラベルとして書き込んでおく．この画像に対して，VLM は，2 節で紹介したタイムシートの記法の下で，左右に描かれている記号と数字を互いの関係性を考慮しながら認識する（4.4 節）．最終的に認識された記号と数字を表計算ソフトや，デジタルタイムシートのファイルフォーマットで出力する．

4.1 事前準備

タイムシートには，1 枚のシート内で記述できる秒数（コマ数）やセル数によってフォーマットにバリエーションが存在する．本研究では，システムに入力される可能性のあるすべての未記入のタイムシートのスキャン画像をテンプレートとして事前に用意し，これをテンプレートプールとする．また各テンプレートのセルの名前（図 1 の場合 A〜G）と ACTION, CELL 欄の列番号も一緒に登録しておく．

また，すべてのテンプレートについて，それらの 2 値画像に対してラベリングを行い，ある面積以上の同じラベルを持つ連結成分について，各領域を画像座標の y 軸方向にのみわずかに拡張した Axis-Aligned Bounding Box (AABB) を計算し，AABB が重なっている領域同士をグループとしてまとめる．グループ内の領域の AABB について各々最

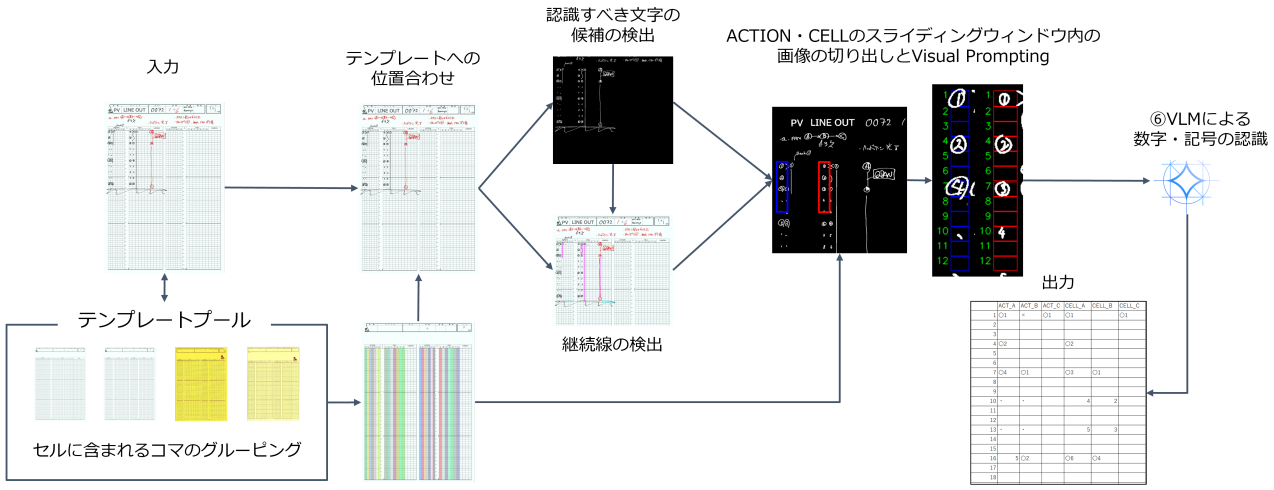

図 2　提案するタイムシートスキャン画像の自動認識フレームワーク

小の y 座標同士を比較し，昇順に並べ替えを行う．これによりセル毎に時系列に沿ったマス目の情報の認識が可能になる．

4.2　テンプレートの選択と入力画像の位置合わせ

テンプレートプールに存在するテンプレートから，入力に最も類似したテンプレートを選択する．入力画像とテンプレート画像から AKAZE 特徴量 [1] を抽出し，特徴点の対応付けを行う．対応する特徴点群からホモグラフィ行列を計算し，対応点のインライア率を入力とテンプレートの類似度とみなし，類似度が最大となるテンプレートを選択する．また，計算されたホモグラフィ行列を用いて，入力画像をテンプレート画像へ位置合わせする．また選択されたテンプレートの領域のグループ情報も同時に取得する．

4.3　認識すべき文字の候補と継続線の検出

位置合わせされた入力画像とテンプレートから認識すべき文字候補と継続線を検出する．位置合わせにより入力画像とテンプレートが整列しているため，差分を取ることでマス目をほぼ除去することができる．しかしながら入力・テンプレートの双方ともにスキャン画像であるため，色むらやノイズが存在し，差分画像をそのまま 2 値化すると，マス目と文字候補や継続線の輝度値が近い場合にそれらが欠損する恐れがある．欠損をできるかぎり抑制するために，差分画像に対してグラブカット [4] を適用して 2 値化する．本稿では経験的に，輝度値が 5 以下を明らかな背景画素，50 以上を文字や継続線の可能性のある画素としてラベルを設定した．

継続線は，上述した 2 値画像に対してラベリングを行い，アスペクト比が 0.5 以上で，高さが 3 コマ分の高さ以上（4.2 節で説明したルールに基づく）となる連結成分のバウンディングボックスとして検出される．検出された継続線のバウンディングボックスと，各セルのマス目のバウンディングボックスとの AABB を計算し，衝突状態にあるマス目については，直前のコマの内容が続くものとみなし，次節の VLM によるマス目に対する認識内容を無視する．

4.4　VLM によるマス目の OCR

タイムシートの全 72 行を一括して処理しようとすると，入力画像のサイズが VLM の想定する最大サイズを超える場合がある．その結果，画像が縮小されて元の情報が欠損し，OCR の精度が下がる可能性がある（この検証結果については次章で述べる）．これに対処するために，ACTION 欄と CELL 欄の双方に重なりのない固定長のスライディングウィンドウを設け，4.3 節で計算した 2 値画像から両ウィンドウ内の画像を切り出し，左右に結合した上で順次 VLM へ入力する．この時，VLM のアテンションを誘導し，マス目の位置や順序の認識をしやすくするための Visual Prompting を施す，具体的には，図 2 に示すように，ACTION 欄の枠を青, CELL 欄の枠を赤，行番号を緑色の文字で描いた後に，認識対象となる 2 値画像の中の文字や記号を含む画素のみを白画素に置き換えた画像を生成する．

OCR の実行時には，紛らわしい文字や記号の認識精度を改善するために，あらかじめ少数の正例を用いた Few-shot Prompting を行う．そして，事前定義した JSON スキーマを強制するプロンプトを用いて，枠内の白色の手書き数字および記号（○，△，×，・）と枠に付与された緑色の行番号を構造化データとして抽出させる．出力された推論結果は，抽出された行番号を介して，タイムシートの元の行にマッピングされる．最後に後処理として，処理対象のマス目のバウンディングボックスと継続線のバウンディングボックスとの交差判定を行う．交差領域の大きさが閾値以上である場合，直前のコマの状態が継続しているものとみ

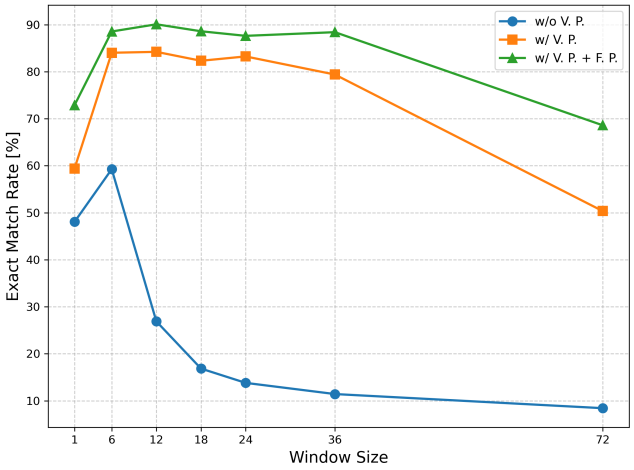

図 3 タイムシート全体での平均完全一致率の比較.

なし，該当するコマにおける VLM の推論結果を棄却する．

5. 実験

本研究では，Gemma 4 26B A4B を VLM として用いた．また，構造化出力のために Outlines[6] を利用した．実際のアニメ制作に使われたタイムシート 30 枚からなるデータセットに対して，スライディングウィンドウのサイズを 1 から 72 まで変化させて提案手法による OCR を行い，手作業での書き起こし結果と比較し，マス目単位の記号・数字の完全一致率 EMR (Exact Match Rate) を，全シートでの平均値として計算した．この際，あるマス目において，OCR と手書きでの書き起こしの結果がともに空欄となっているマスについてはカウントから除外している．提案手法の要素ごとの寄与を確かめるために，Visual Prompting の有無（w/o V. P. vs w/ V. P.），Few-shot Prompting の有無（w/ V. P. vs w/ V. P. + F. P.）による比較も行った．なお，Visual Prompting を行わない場合については，行番号は描かず，ACTION, CELL 欄ともにグレーの枠のみを下地としてレンダリングすることとした．結果を図 3 に示す．提案するフレームワークは，ウィンドウサイズを 12, Visual Prompting, Few-shot Prompting を共に設定した状態で，平均完全一致率 90.1%を達成している．

図 3 を見ると，スライディングウィンドウのサイズが 72 のとき，すなわち，タイムシートの 1 列すべてを一度に OCR しようとした場合に全てのケースにおいて EMR が著しく低下していることが分かる．これは，入力画像が VLM の想定する入力画像のサイズを超え，画像がダウンサイズされることによる情報の欠損によるものと考えられる．このことから，細かいブロックごとに順次 OCR していくことが有効であることが伺える．一方で，スライディングウィンドウのサイズを 1 としたときも同様に EMR が低下した．この結果は，タイムシートの認識において前後のコマの文脈を考慮することの有効性を示唆している．

Visual Prompting を追加した場合，ウィンドウのサイズが大きくなるほどその効果が表れており，マス目の順序の認識において Visual Prompting が重要な役割を果たしていることが伺える．加えて Few-shot Prompting の導入により，より安定した，高精度な認識を実現できていることが，この結果からは見て取れる．

ウィンドウサイズは，小さくすればするほど推論コストの高い VLM の呼び出し回数が増えてしまうため，適切なサイズを設定する必要がある．本実験で使用したタイムシートでは，ウィンドウサイズが 6 の時，平均 155 秒,18 の時に 73 秒（NVIDIA RTX Pro 6000, 96GB）を処理時間に要する．適切なサイズを決定するために，今後より大規模かつ，様々なバリエーションを持つタイムシートに対する実験を行う必要がある．

6. おわりに

本論文では，決定論的な CV 技術と確率的な VLM を組み合わせたタイムシートスキャン画像に対する OCR フレームワークを提案した．提案したフレームワークは，実際に制作に用いられた小規模なタイムシートのデータセット上で，平均完全一致率 90.1%を達成した．今後の課題として，より大規模かつフォーマットのバリエーションをともなったタイムシートのデータセット上で，提案するフレームワークの評価を行う必要がある．

謝辞

本研究は，JST，CRONOS，JPMJCS25K1 の支援を受けたものです．

参考文献

[1] Alcantarilla, P. F., Nuevo, J. and Bartoli, A.: Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces, *Proceedings of the British Machine Vision Conference (BMVC)*, BMVA Press, pp. 13.1–13.11 (2013).

[2] CELSYS, I.: 東映アニメーション デジタルタイムシート (2018). アクセス日：2026.6.5.

[3] Li, M., Lv, T., Chen, J., Cui, L., Lu, Y., Florencio, D., Zhang, C., Li, Z. and Wei, F.: TrOCR: transformer-based optical character recognition with pre-trained models, *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'23/IAAI'23/EAAI'23, AAAI Press (2023).

[4] Rother, C., Kolmogorov, V. and Blake, A.: GrabCut: Interactive foreground extraction using iterated graph cuts, *ACM transactions on graphics (TOG)*, Vol. 23, No. 3, ACM, pp. 309–314 (2004).

[5] Smock, B. and Pesala, R.: Table Transformer (2021).

[6] Willard, B. T. and Louf, R.: Efficient Guided Generation for Large Language Models, *arXiv preprint arXiv:2307.09702* (2023).