SCVeamer PDF処理の置換試作

pypdf + pypdfium2 / PDFium の評価 | 2026-10-24（日本時間）

結論：第一候補として進める見込みがある

PyMuPDFを含まない環境で、公開キットの既存PDFから、段階表示とゲームを含む単一HTMLを生成できた。日英の構造・タイトル・時刻・固定例と、配布用PDFのページ選択を維持した。通常表示の画質は良好だが、拡大時の鮮明さと閲覧メモリにはSVG方式との差が残る。 この試作を評価し、実機確認後に公開版への統合を判断する。

比較条件

SCVeamer 1.3の日英公開ZIPを取得し、引継ぎ記録のSHA-256と一致を確認。Linux / Python 3.12.14で、旧PyMuPDF 1.26.6と新方式を別の依存環境で実行した。新方式はpypdf 6.10.0、pypdfium2 5.14.0、Pillow 12.3.0を固定。PDFiumは156.0.8076.0。元の公開サイトと原本は変更していない。

Day1（46枚 / 76段階）	HTML容量	変換時間	巡回後PSS
現行 SVG（日本語）	18.52 MB	1.23 秒	670.2 MiB
PNG 1920px（日本語）	23.36 MB	9.36 秒	1011.8 MiB
PNG 2560px（日本語）	28.86 MB	14.41 秒	1048.5 MiB
現行 SVG（英語）	15.04 MB	0.99 秒	未測定
PNG 2560px（英語）	28.44 MB	13.28 秒	1042.4 MiB

容量のMBは1,000,000バイト。変換時間はHTML生成関数の各1回の実測で、初回導入・handout生成を含まない。PSSはChromiumの関係プロセス合計の一時点の値で、ピーク値ではない。実機・資料内容・キャッシュ条件により変わる。

今回の成果物

日英のソースキット、横2560pxで生成した日英Day1の単一HTML、各キット内のスターター・テンプレート・配布用PDF、固定依存一覧、第三者ライセンス類、検証記録を用意した。同梱資料の設定も2560pxに揃えた。解像度はCLIまたはpresentation.jsonで変更できる。

採否の考え方

1080p前後での講義を想定した候補として試せる。1920pxは容量を抑え、2560pxは拡大の余裕を増やす。高倍率でもベクターと同じ鮮明さを必須にする場合、またはメモリ増加が実機で支障になる場合は、次段階でPDF.js等を小規模に比較する。

機能・操作・配布PDFの確認

項目	今回確認した結果
構造・目次	日英ともスターター6枚/9段階、テンプレート9/13、Day1 46/76。全フレームのタイトルと機能データが一致。
ゲーム・時刻	Day1の8つの固定例、3ゲーム、言語・数式表示、14:55開始/135分/17:10終了を維持。
ブラウザ	Linux Chromium 156のfile://閲覧、通信無効。4条件で各103項目成功。全76段階、実際のスライダー操作と状態保持、iframe再利用、印刷用画像を確認。
handout	日英Day1は46ページ。選択ページ・内容ストリーム・寸法・残すリンクが一致。全46ページの512px描画も元PDFの該当ページと一致。PNGをPDFへ貼り直していない。
テスト・入力保護	日英とも17件成功。既存15件の意図を保持し、注釈の選別と段階表示の隠れた文字に対する2件を追加。原稿・元PDFを維持。

メモリの小規模改善も試したが、今回は採用しない

全画像の先読みを止め、現在と前後の表示だけを保持する実験では、巡回後PSSは1048.5→1025.3MiB（約2.2%減）だった。一方、初期・印刷後のメモリや移動時間は一貫して改善しなかった。1回ずつの測定でこの差を効果と判断せず、同梱プレイヤーは元版のままにした。

文字抽出は表示画像と分けて扱う

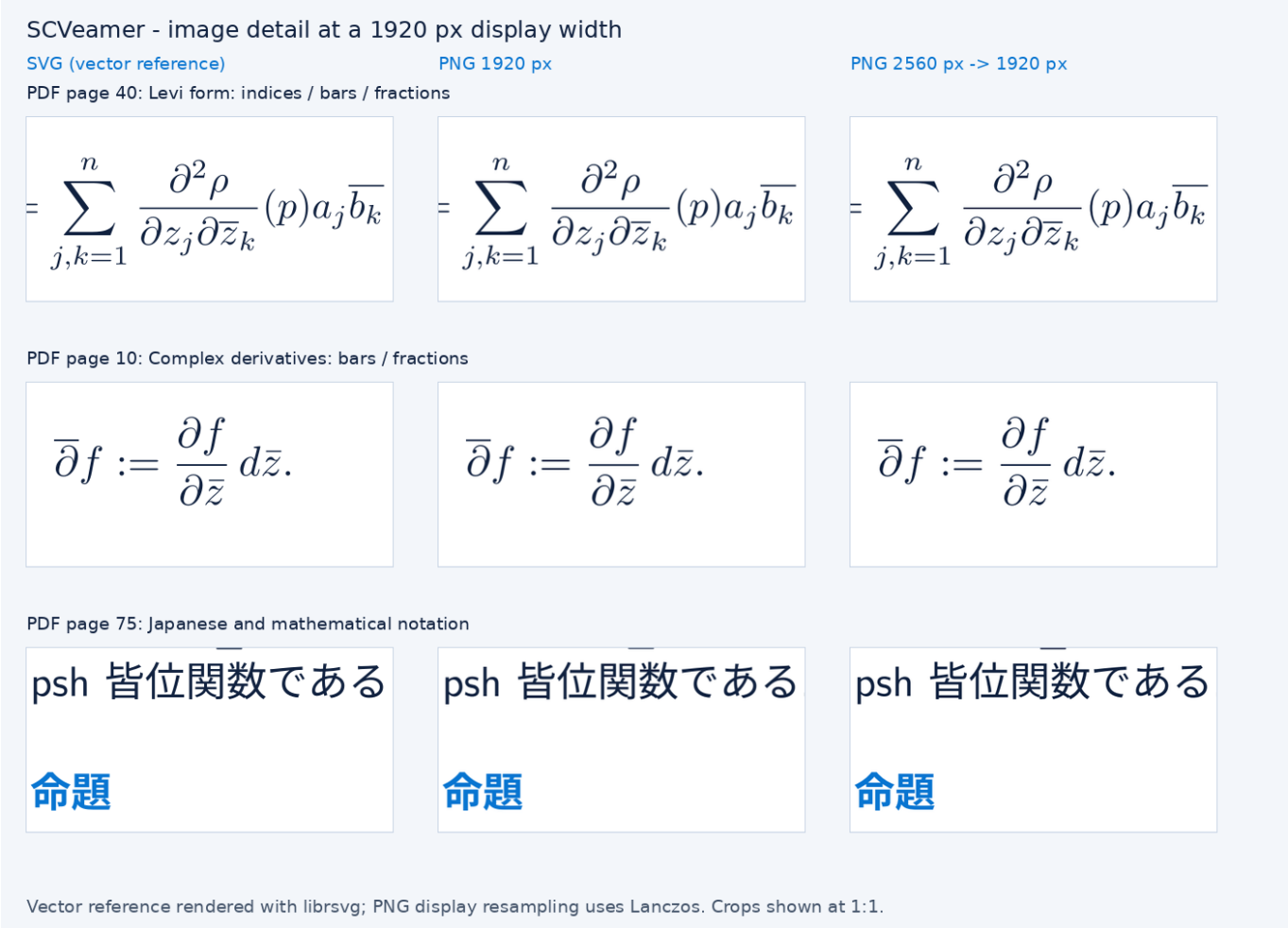
pypdfで本文を直接抽出すると、Beamerがページ外へ移して隠している後続段階の文章まで拾う問題を発見した。画像のaltテキストにはPDFiumのページ範囲内抽出を使い、回帰テストで修正を確認。上付き記号のあるタイトルは別に抽出し、実サンプルの全タイトルは一致した。ただしalt内のプライム・否定線・証明終了記号などには文字対応の差が残る。これはPNG内の数式の描画とは独立である。

実施範囲の限界

Windows/MacのGUI、初回導入、exe化、会場での投影、任意の回転・CropBox・暗号化PDF、TeXからの一括コンパイルは今回の実機確認に含めていない。TeX一括生成の機能は残している。セキュリティ監査や大学等の権利確認を完了したという意味ではない。

画質比較：横1920pxで表示

同じPDFページの同じ部分を、横1920px表示に揃えて比較。日本語・英語、細い分数線、添字、上線、図の線を代表ページで確認した。確認範囲では、本文の欠落・文字化け・数式の組版崩れは見られなかった。

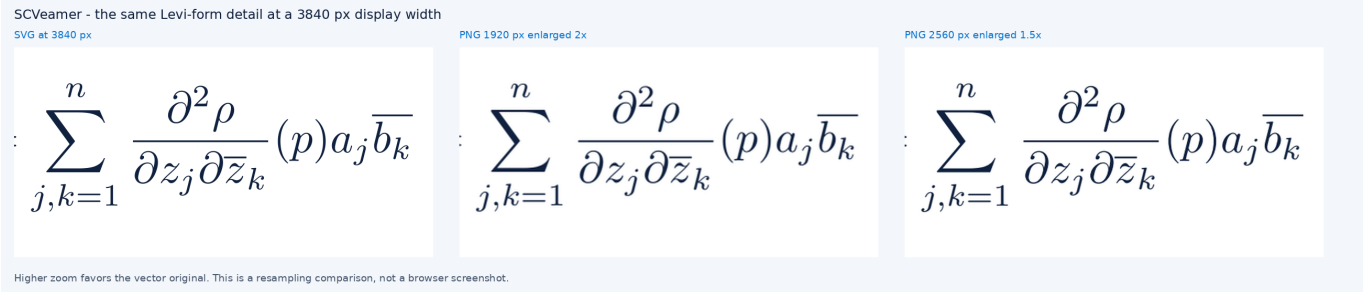


左：現行SVGを指定解像度で描画。中央：1920px PNG。右：2560px PNGを縮小。比較画像のSVG描画はlibrsvg、PNGの縮小はLanczosを使用しており、ブラウザの完全な描画差分ではない。元の数式・講義内容は変更していない。

PNGのキャンバスは透明を保持し、PDF自体が塗っていない部分にはプレイヤーの紙色が使われる。同梱PDFの寸法とPDFiumの切上げにより、実際の画像寸法は1920×1081pxまたは2560×1441pxとなる。

拡大表示と、次に確かめること

横3840px（4K相当）へ拡大すると、PNGはSVGより輪郭が柔らかくなる。2560px版は1920px版より余裕があるが、任意の倍率で鮮明なベクター方式と同一ではない。



手元での確認は完成HTMLを開くところから

まず日英いずれかのDay1 HTMLを保存して、普段使うEdgeまたはChromeで開く。ネット接続を切った状態でもページ送りとゲーム操作ができるかを確かめる。数式の多いLevi形式のページや小さな添字を、普段の全画面・拡大率・投影条件で見る。ゲームを操作して前後へ移動し、状態が残ることも確認する。

再生成する場合

キットを新しいフォルダへ展開し、Python 3.11以上を用意する。SETUP_WINDOWS.cmdで専用環境を準備した後、GUIの「既存PDFからHTMLを作成」、または次のCLIを使う。初回のライブラリ取得にはネット接続が必要だが、完成HTMLの閲覧には不要。

```
py scveamer.py --project "C:/work/my-talk"
--pdf-only --image-width 2560
```

上記コマンドは1行で実行する。PDFだけからの変換にTeXは不要。既存の講義原稿で試すときは、資料フォルダの作業用コピーを指定する。詳しい手順はキット内 README_PDFIUM_PROTOTYPE.txt。

権利表示と依存部品

第三者部品の既存表示を保持し、使用したwheelに付属する20件のライセンス文書を収録した。pypdfium2/PDFiumは、ビルドに含まれる第三者部品の条件も確認対象になる。今回の収録物は検証したLinux用パッケージ由来であり、Windowsバイナリを作る場合にはその実物の表示を確認する。自作コードの新たなMIT宣言や権利者の確定は行っていない。

一次資料・再現情報

公開原本：SCVeamer配布ページ
公式仕様：pypdf 6.10.0 ／ pypdfium2 API ／ pypdfium2 / PDFiumのライセンス説明
取得日・ハッシュ、比較JSON、テスト・ブラウザ記録は各キットのREFERENCE_SOURCES.json、VALIDATION_PDFIUM.txt、validation/に収録。