TL;DR TikZ lets you script motion graphics like equations, data charts, and UI wireframes in LaTeX, then export frame stacks that stay razor sharp at any resolution. Combine latexmk, ImageMagick, and ffmpeg to automate the full video handoff.

Why bring LaTeX and TikZ into video production?

- Procedural control: drive every vertex, curve, and easing function directly from formulas or CSV feeds.
- Brand precision: reuse the same typographic system your team uses for decks and PDFs, no font swapping in post.
- Resolution agnostic: render vector frames to 4K, 8K, or infinity without redoing layout.
- Repeatability: version control the scene with Git, build variants per language or product, and drop the results into your existing creative ops pipeline.

Typical workflow

1. Storyboard the motion beats and call out reusable TikZ components (graphs, diagrams, pointer paths).
2. Write a LaTeX document that parameterises frame counts, colours, and easing curves.
3. Use a build script to render sequential PDFs or SVGs with `latexmk`.
4. Convert to image frames (PNG/WebP) or keep SVG for vector-friendly editors.
5. Assemble the final clip with `ffmpeg`, or hand the sequence to Premiere, After Effects, or your automated generator.

Minimal animated scene

```
% file: orbit-animation.tex
\documentclass[tikz]{standalone}
\usepackage{tikz}
```

```
\usepackage{animate}

\begin{document}
\begin{animateinline}[poster=first,controls,autoplay]{12}
  \multiframe{60}{rAngle=0+6}{%
    \begin{tikzpicture}[scale=3]
      \draw[very thin, gray!40] (0,0) circle (1);
      \fill[blue!60] (\rAngle:1) circle (0.07);
      \draw[->, thick] (0,0) -- (\rAngle:1);
      \node[anchor=west] at (1.1,0.3) {Angle: $\rAngle^\circ$};
    \end{tikzpicture}%
  }
\end{animateinline}
\end{document}
```

Build the sequence to PDF frames:

```
latexmk -pdf -interaction=nonstopmode orbit-animation.tex
```

The `animate` package embeds all 60 frames in a single PDF. To export the sequence for video work, split the animation into page-based frames.

Exporting frames to video

```
# 1. Burst the PDF animation into individual pages
pdfseparate orbit-animation.pdf frames/frame-%03d.pdf

# 2. Convert each page to transparent PNG (or SVG if you prefer vector in post)
for f in frames/frame-*.pdf; do
  magick -density 400 "$f" -background none "${f%.pdf}.png"
done

# 3. Assemble into a 12 fps mp4 clip
ffmpeg -framerate 12 -i frames/frame-%03d.png \
  -c:v libx264 -pix_fmt yuv420p exports/orbit.mp4
```

Tips:

- Increase `-density` for higher pixel density when rasterising.
- Add `-vf scale=1920:1080` in `ffmpeg` if you need a specific canvas.
- Encode WebM versions in the same loop for HTML or lightweight previews.

Making LaTeX scenes data-driven

TikZ happily reads external data, so you can generate frames from CSV or JSON exports.

```
\foreach \row [evaluate=\row as \value using {\pgfplotstableread{data.csv}\row\value}] in {0,...,59} {
  % Use \value to position bars, text, or node colours per frame
}
```

Combine with a short Python script that rewrites the data file per render run to bake in A/B scenarios or localisation strings.

```
from pathlib import Path
import csv

def write_data(values):
    with open("data.csv", "w", newline="") as fh:
        writer = csv.writer(fh)
        writer.writerow(["frame", "metric"])
        writer.writerows(enumerate(values))

write_data([ease_in_out(t / 60) for t in range(60)])
```

This approach keeps analytics or scientific visuals perfectly synced with the narration in your video timeline.

Integrating with production tools

- **After Effects or Premiere:** Import the PNG sequence; both apps auto-detect numbered files. Use the vector PDF if you rely on third-party scripts that preserve Illustrator fidelity.
- **Programmatic pipelines:** Wrap the commands above in a Node or Python task so your render farm can output new clips on demand (e.g., weekly reports or personalised onboarding videos).
- **Brand systems:** Centralise colours, fonts, and line styles in a shared LaTeX package. Update once, roll out everywhere.
- **Audio sync:** Keep a JSON manifest of frame timestamps to align VO cues or SFX during video assembly.

Production notes

- Watch out for heavy shadows or gradients: rasterisation is fast, but embellishments slow down ImageMagick. Consider exporting to SVG for vector-based editors when possible.

- Use `latexmk -jobname=frame%03d` tricks if you want each frame generated directly, avoiding `pdfseparate`.
- Version control your `.tex`, shared packages, and build scripts so the creative and engineering teams can trace changes.
- Add QA hooks: diff a pair of rasterised frames to spot unintended changes before shipping to clients.

Need a hand folding TikZ animation into your motion graphics pipeline?

References

- PGF/TikZ (GitHub)
- TikZ-Feynman (arXiv)
- ManimCommunity (GitHub)
- Manimator (arXiv)