TIK: a time domain continuous imaging testbed using conventional still images and video

Henry Dietz, Paul Eberhart, John Fike,
Katie Long, Clark Demaree, and Jong Wu

DPMI-081, 11:30AM February 1, 2017

University of Kentucky
Electrical & Computer Engineering

University of Kentucky

The Philosophy of Time Domain Continuous Imaging (TDCI)

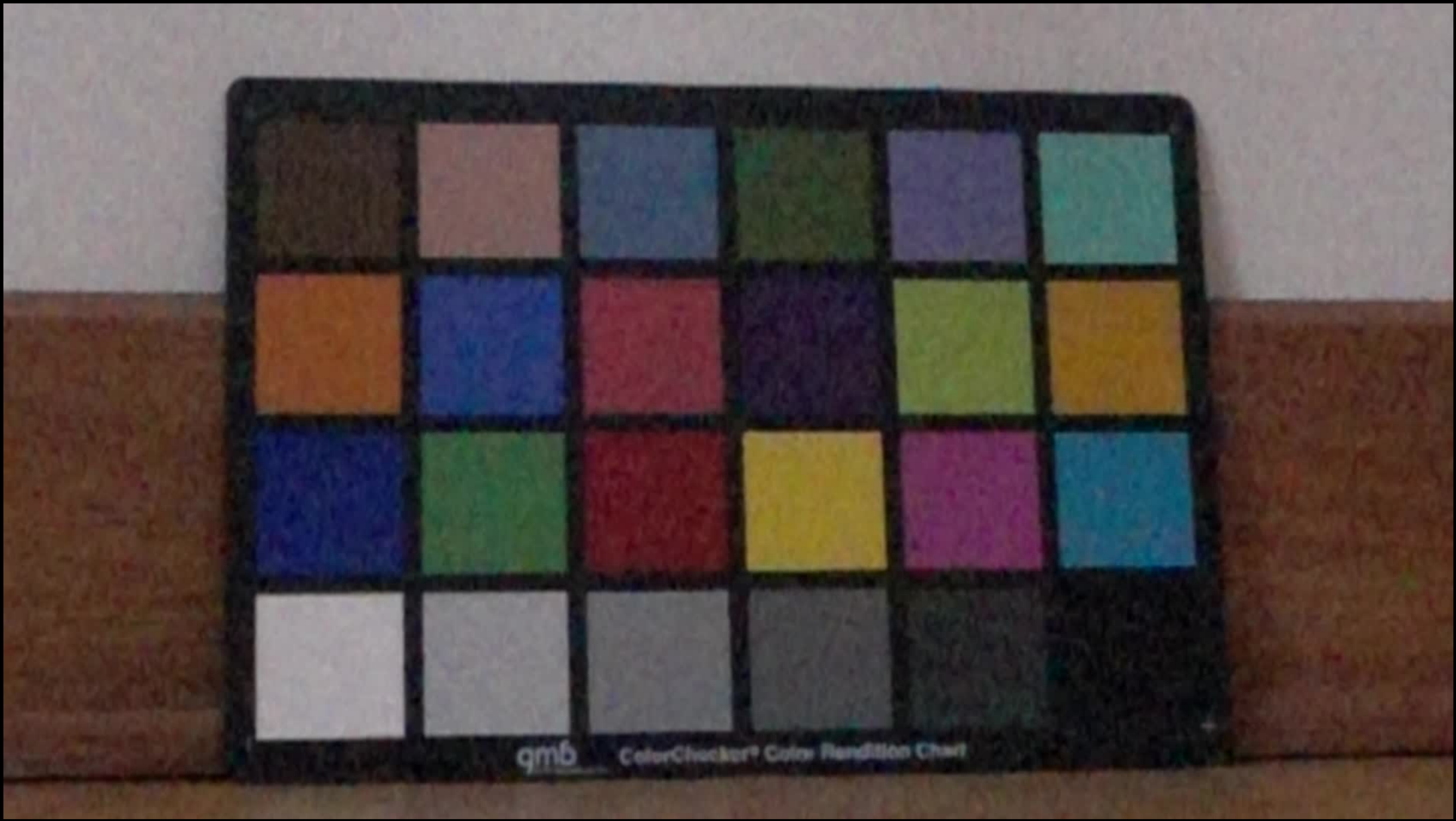
- Cameras create scene appearance models
- Photons are the sampling mechanism
- Thought experiments:
  - What is the value of a pixel in T=[1..2] if in T=[0..3] photons hit at T=0.99, T=2.01?
  - What color is a blue sheet of paper when no photons are sampling it?
- Scene appearance model changes as a (mostly) continuous function over time

University of Kentucky

What TDCI Can Do

- TDCI representation: a continuous waveform per pixel, compressed (mostly) in time domain
- TDCI processing enables:
  - High dynamic range (HDR), improved SNR
  - Rendering a virtual exposure for any time interval (start time, shutter speed)
  - Rendering a conventional video at any FPS and shutter angle (temporal weighting)

A 960FPS Video Frame



University of Kentucky

A 1/960s Virtual Exposure

University of Kentucky

A 1/24s Virtual Exposure

University of Kentucky

A 1s Virtual Exposure

University of Kentucky

TIK

- TIK stands for:
  Temporal Image Kontainer
  (or **Temporal Imaging from Kentucky***)*
- TIK is a specification of TDCI file formats
  - Simple TDCI formats extending PGM/PPM
  - Extensions of DNG for handling “raw” TDCI (not yet implemented)
  - A format for specifying a value error model
- File names should end in **`.tik`**

University of Kentucky

TIK File Metadata

 # TIK B *number*

Delay in ns to when capture began

 # TIK E *number*

Approximate EV for scaling luminances

 # TIK F *number*

Frame time in ns (1G/FPS)

 # TIK G *number*

Approximate gamma

 # TIK R *number numberXdiv numberYdiv*

Rolling shutter timing specification

TIK File Metadata

- **`# TIK T`** *`number`*
  Shutter open time in ns
- **`# TIK V`** *`version format ...`*
  Version compliance date and format
- **`# TIK X`** *`number`*
  X dimension (width) of the image data; each RGB or UYVYYY counts as one unit
- **`# TIK Y`** *`number`*
  Y dimension (height) of the image data
- **`# TIK Z`** *`number`*
  Maximum value of a color channel

University of Kentucky

TIK File Formats

`20160721 CONVERT` *`pattern numBegin numEnd`*

Stills fed by ImageMagick **`convert`**

`20160721 FFMPEG` *`filename`*

Video fed by **`ffmpeg`**

`20160712 RGB`

Spatio-temporal TDCI with P6 PPM header

`20160712 UYVYYY`

Spatio-temporal TDCI from CHDK PowerShot

`20160804 NOISE`

P6 PPM pixel value error model

University of Kentucky

TIK

- Currently, no sensors directly implement TDCI

- `tik` is an open-source C program
- `tik` can derive a TDCI model of scene appearance from timestamped images:
  - Still images with temporal annotation
  - Video frames with FPS, shutter angle
- `tik` can construct/use a **value error model**
- `tik` can render **virtual exposures**, **video**

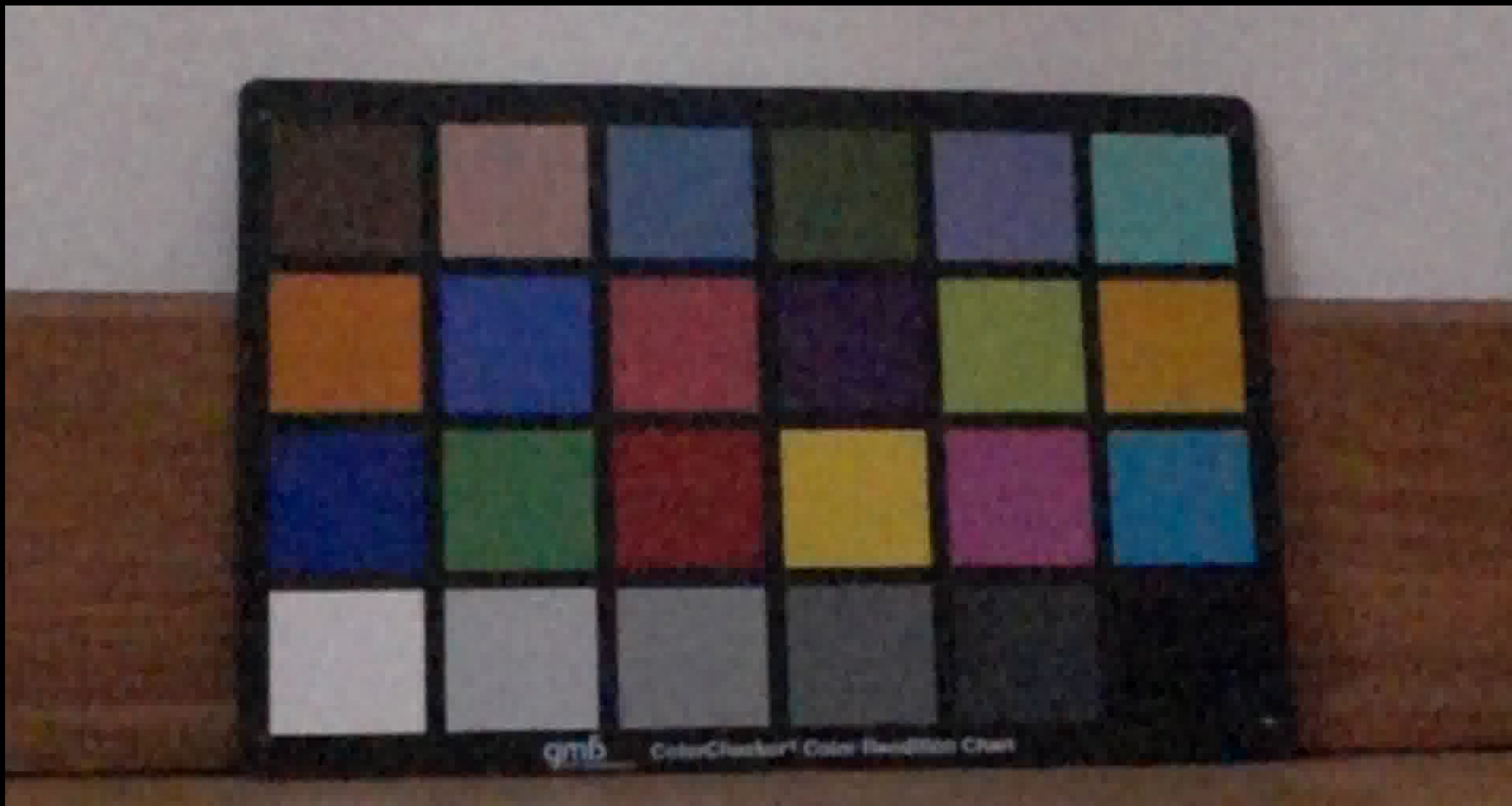
University of Kentucky

Creation of an Error Model

1. Capture video or still sequence of a completely static scene
2. For each pixel value in each color channel, create a histogram of values it transitions to
3. Convert the histogram into probabilities scaled into the range [0..255]
4. Make monotonically non-decreasing
5. Create 256x256 map as X is previous value, Y is next value, RGB are scaled probabilities

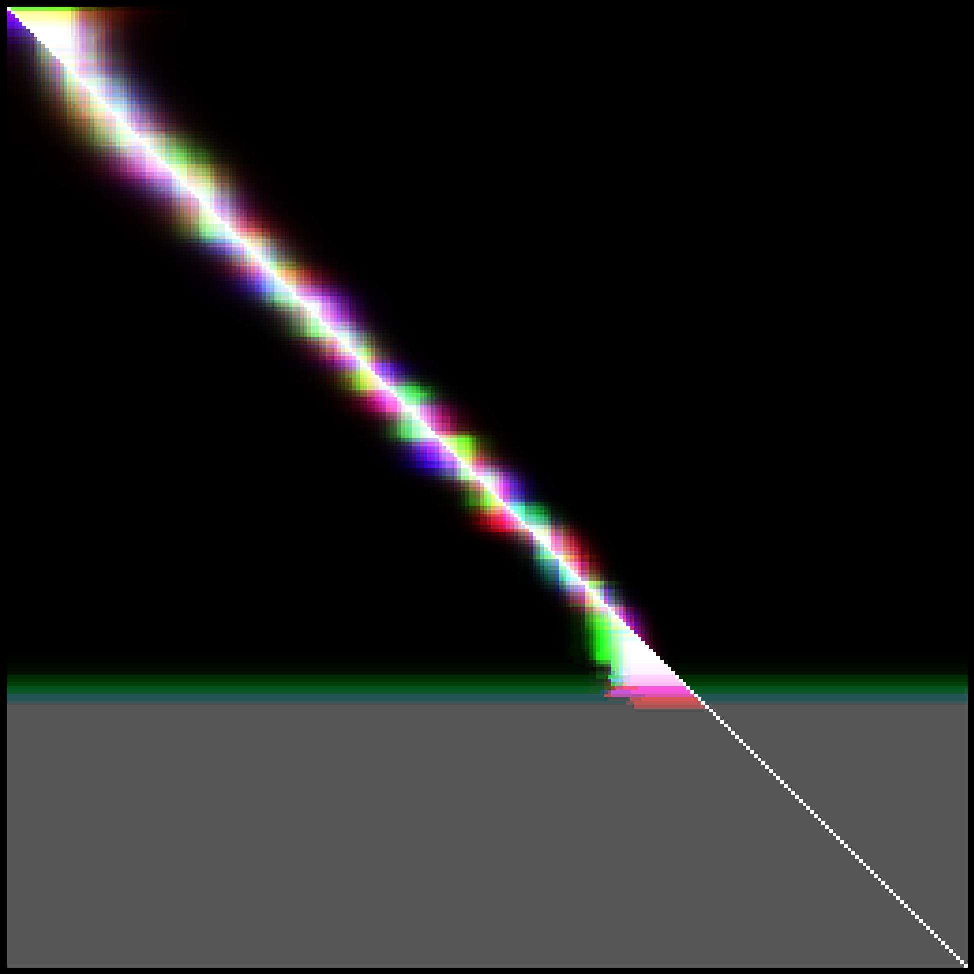
University of Kentucky

960FPS from Sony RX100 IV



University of Kentucky

Empirical Noise Map for 960FPS Sony RX100 IV Video



University of Kentucky

Creation of RGB `.tik`

1. Examine a video frame / still image at a time
2. Scan each image (in roll order, if specified)
3. Check if R,G,B pixel value is same within error model as previous change record for this pixel
4. If pixel changed, write pixel-change record: {spatio-temporal distance to last change record for any pixel, new pixel value};
   If not, improve previous change record value

Rendering Virtual Exposures

1. Start scanning pixel value change records at beginning of TDCI, tracking current waveform value for each pixel
2. When pixel record time is in virtual exposure interval, integrate under curve
3. Continue scan until all pixels are past interval
4. Map exposure range to output pixel range and output frame
5. Repeat for each virtual exposure

Original Video Capture

- Original 240FPS capture at 1/250s (~346°)

Virtual Video Sequence

- Virtual 240FPS video (original rate) with 360°

University of Kentucky

Virtual Video Sequence

- Virtual 24FPS video with 360°

University of Kentucky

Virtual Video Sequence

- Virtual 100FPS video with 360°

Original Video Capture

- Original 240FPS capture at 1/320s (270°)

University of Kentucky

Virtual Video Sequence

- Virtual 240FPS video with 360°

University of Kentucky

Virtual Video Sequence

- Virtual 25FPS video with 90°

Virtual Video Sequence

- Virtual 29.97FPS video with 360°

University of Kentucky

Virtual Video Sequence

- Virtual 300FPS video with 360°

Conclusions

- TDCI significantly improves upon frame-based imaging – even using conventional sensors
- TIK testbed: **`.tik`** formats + **`tik`** software
- Lots of room for improvement:
  speed, interpolation algorithms, "raw," etc.

Aggregate.Org
UNBRIDLED COMPUTING

University of Kentucky