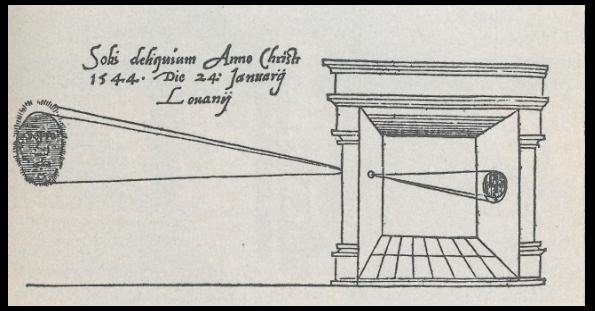
Introduction to Digital Photography

May 17, 2021

Henry Dietz
http://aggregate.org/hankd

University of Kentucky

Evolution of Photography



- Camera obscura
- Film
  - Silver halide ⇒ metallic silver
  - Expensive, requires chemical processing
  - Single use, but creates a unique original
- Digital using CCD or CMOS sensor
  - Sensor is ~linear analog photon counter
  - Images become digital computer data

Digital Imaging

- Digital images allow
  - Various storage media are cheap & reusable
  - Lossless copy, permanence (colors don’t fade)
  - Immediate digital processing & display – even producing a live view
- Computer control allows “smart” imaging
  - Processing of the image data
  - Control over capture parameters

Shutterless Digital Cameras

- Webcam: no mechanical shutter; video oriented; usually low cost, resolution, & image quality (IQ)
- Camcorder/DV: video oriented; low end units are webcams that don't need to be plugged in
- Cell: tiny camera module(s) in a phone; IQ is inherently low, but extensive processing helps
- Industrial/machine-vision: rugged webcam with a better sensor and interchangeable lens

Midrange Digital Cameras

- Compact: fits in a pocket, limited manual control
- Waterproof/rugged: “weather sealed” compact
- Super-Zoom: built-in lens covers wide range of focal lengths, may compensate for shake

Larger-Sensor Digital Cameras

- Prosumer: compact with bigger sensor, higher IQ, manual controls, lots of “creative options” for use
- DSLR: (Digital Single Lens Reflex) deflects view through the lens to optical viewfinder (OVF) using a movable mirror; interchangeable lenses
- Mirrorless/EVIL: electronic viewfinder (EVF) shows view through the interchangeable lens

Exposure

- How much light the sensor sees
- A function of 4 things:
  - Available light... which is hard to control
  - Shutter speed
  - Aperture or f/number
  - “Film” speed (quantum efficiency, etc.)
- Generally, if available light is constant, other parameters trade off

Shutter Speed

- Duration of period during which light is sensed
- 2X time is 2X light energy
- Speeds from about 30s to 1/8000s:

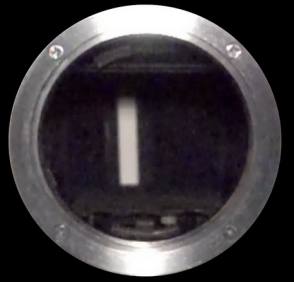
```
30 15 8 4 2 1 1/2 1/4 1/8 1/15 1/30
1/60 1/125 1/250 1/500 1/1000 1/2000
1/4000 1/8000
```

Note decimal-centric rounding....

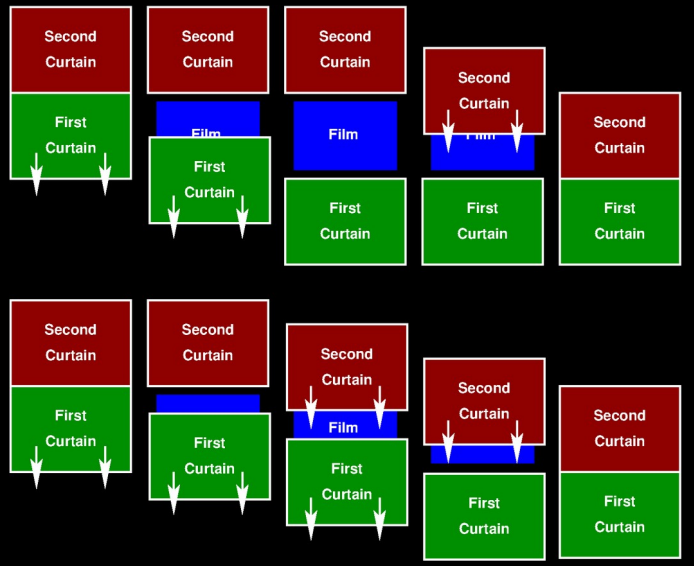
Shutter Speed

- Things moving faster than the shutter blur (that's everything if you move the camera)
- Longer than 1/30s or 1/focal_length, brace the camera (use a tripod, lean on something, etc.)
- Anti-shake can help
  - Optical Image Stabilization (OIS)
  - In-Body Image Stabilization (IBIS)
  - Multi-shot/computational anti-blur

Shutter Mechanisms



- Leaf – compacts
- Focal-plane – DSLR, EVIL
- EFCS (Electronic 1st Curtain)
  – EVIL avoids close, reopen
- Rolling electronic – common
- Global – some industrial



Aperture or f/number

- How much light is admitted by the lens (T/number is light transmitted by the lens)
- Larger aperture is smaller f/number; f/number = focal length / aperture diameter
- Light goes as $1/f^2$; in 2X steps:

`0.7 1 1.4 2 2.8 4 5.6 8 11 16 22 32`

Rounding of powers of sqrt(2)....

Film Speed: EI, ISO, ASA

- Quantum efficiency (QE) isn't alterable, but can change analog (ADC) and digital "gain" factors
  - Analog gain can increase noise, might clip
  - Digital gain better if "ISOless" or "ISO invariant"
- Higher is more sensitive; in 2X steps:

```
102400 51200 25600 12800 6400 3200
1600 800 400 200 100 50
```

Film Speed & Noise

ISO 50 vs. ISO 400 (on an old camera)

Shutter Speed

- Duration of period during which light is sensed
- 2X time is 2X light energy
- Speeds from about 30s to 1/8000s:

```
30 15 8 4 2 1 1/2 1/4 1/8 1/15 1/30
1/60 1/125 1/250 1/500 1/1000 1/2000
1/4000 1/8000
```

Note decimal-centric rounding....

APEX Exposure Computation

Additive System of Photographic Exposure

$$Av + Tv = Bv + Sv = Ev$$

Av	Aperture value; if f/1.0=0, f/1.4=1
Tv	Time value; if 1s=0, 1/2s=1
Bv	Brightness value; if 1fL=0, 2fL=1
Sv	Speed value; if ISO100=5, ISO200=6
Ev	Exposure value for judging equivalence

PASM Modes

- Program: camera picks shutter and aperture

$$\textbf{if } Ev_{target} - Av_{min} < Tv_{blur} \textbf{ then } \{ Av = Av_{min} ; Tv = Ev_{target} - Av_{min} \}$$
$$\textbf{else if } Ev_{target} - Tv_{blur} < Av_{limit} \textbf{ then } \{ Tv = Tv_{blur} ; Av = Ev_{target} - Tv_{blur} \}$$
$$\textbf{else } \{ Av = Av_{limit} ; Tv = Ev_{target} - Av_{limit} \}$$

- Aperture: user picks aperture
- Shutter: user picks shutter speed
- Manual: user picks aperture & shutter speed
- Auto ISO is usually a separate selection...

Lens Focal Length

- Shorter means wider viewing angle
- Sensor size varies; use 135 format equivalent:
  - 135 is 36x24; Sqrt($36^2 + 24^2$) ≈ 43.27mm
  - Wide angle is <40mm (e.g., 35mm)
  - Normal is 40-58mm; “fast fifties” are common
  - Telephoto is >58mm (e.g., 85mm)... technically telephoto means “lens shorter than focal length”
  - Ultrawides as short as 9mm

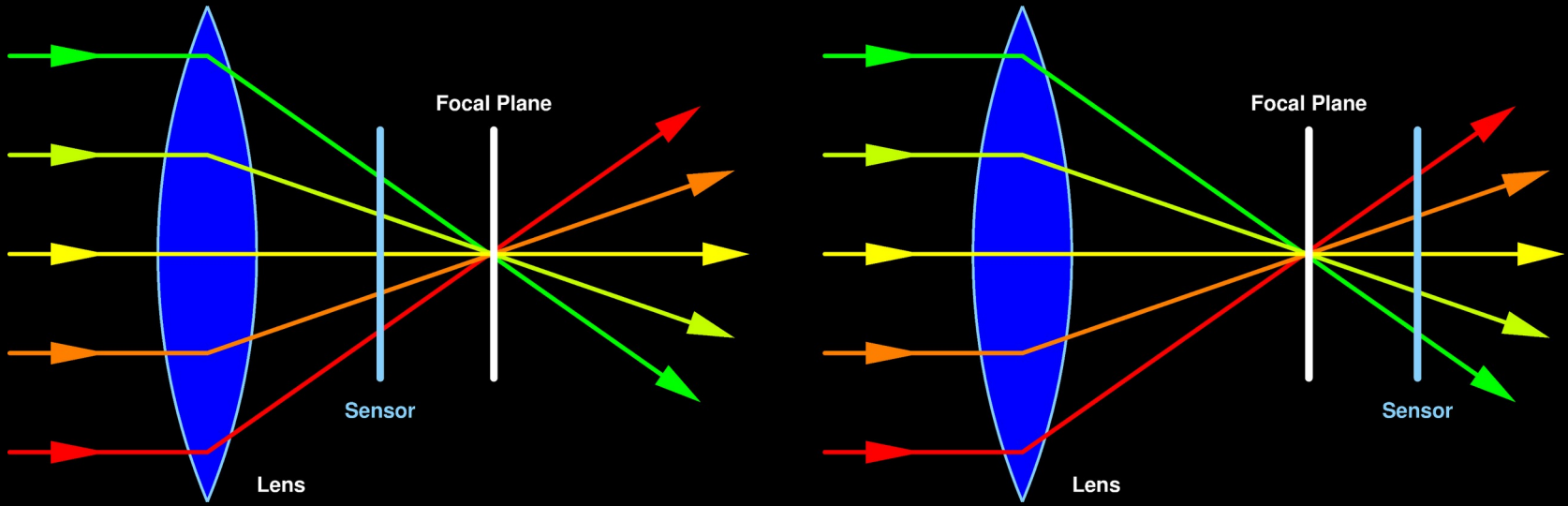
Lens Depth of Field (Focus?)

- Depth of Field (DoF):
  - Distance range that is in sharp focus
  - Smaller focal length increases range in focus
  - Higher f/number increases range in focus
- Depth of Focus (also DoF):
  - How far off from image plane is still in focus
  - Film emulsions and flatness problems limited resolution; sensors are very thin and flat

Lens Depth of Field

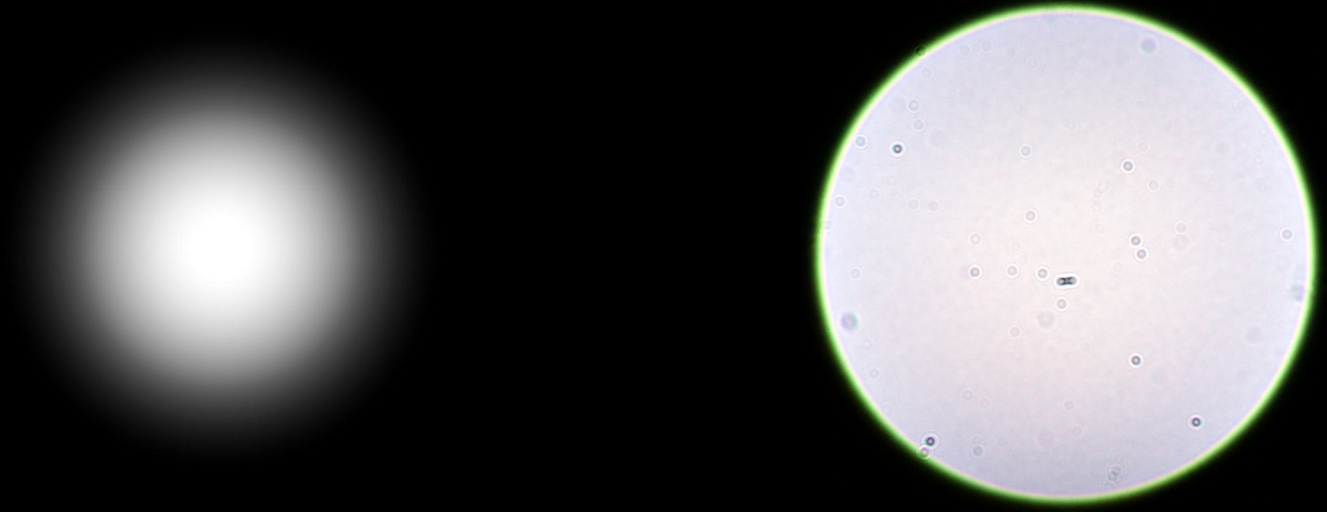
7.0mm (36mm) f/8 vs. 20.3mm (104mm) f/2.5

What The Aperture Does



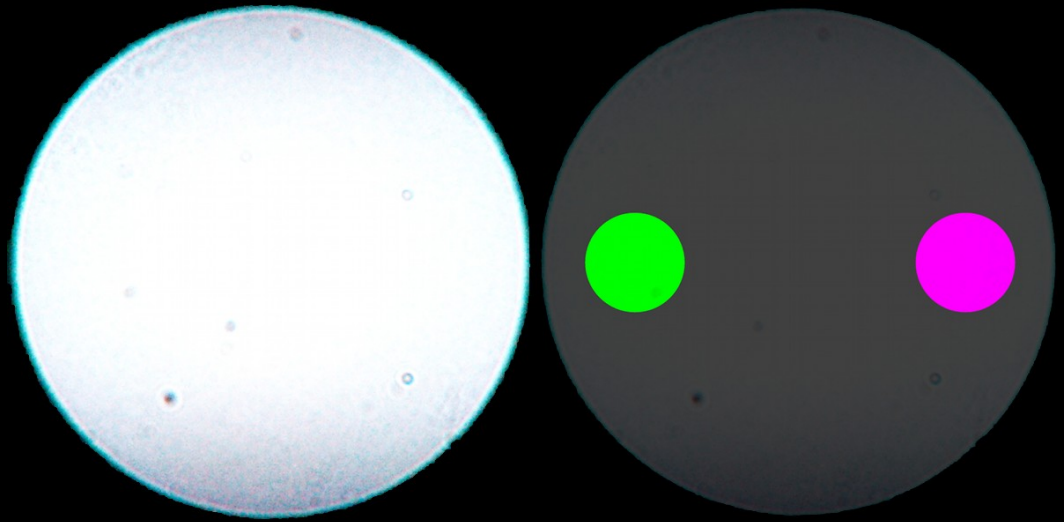
A larger aperture lets more rays pass from each point in the scene, but each ray is from a slightly different point of view

What The Aperture Does



Out-of-focus (OOF) image of a point of light
is the OOF point spread function (PSF)...
and **it isn't blurry!**

Example Using The OOF PSF



Can shape (e.g., color-code) the aperture to directly capture an Anaglyph image distinguishing left and right viepoints... hence, encoding object distance

Flash (Strobe)

- A light pulse synchronized with the shutter
  - Gives fast exposure in poor ambient lighting
  - Efficient; high energy is output only when needed
  - Limited useful range, tends to look “flat”
- Red Eye and red-eye reduction modes
- Fill-in flash and flash at slow shutter speeds
- Bounce or otherwise soften flash lighting

Fill-in Flash Example

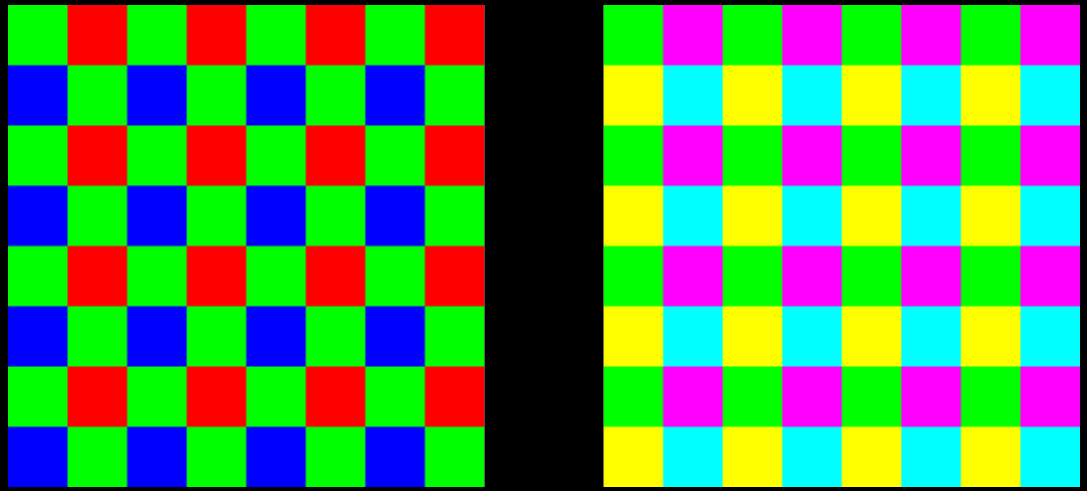
Used to flatten harsh shadows, backlighting

Image Capture

- Sensors are basically analog photon counters, digitized to 8-16 bit linear digital value per pixel
  - CCD reads out via “bucket brigade”
  - CMOS random access, a lot like DRAM
- 135 film quality is ~1.5MP to 6MP digital
- Grain is noise and pixel count
- Noise reduced by more photons, bigger pixels, colder temperatures, and faster shutter speeds

How Sensors See Color

Color Filter Array (CFA) on sensor



EOS-1D, GRBG Bayer Pattern vs. G1, GMYC

Overexposure vs. Underexposure

Over clips; Under simply decreases SNR

Shutter Speed and Noise

Film has reciprocity failure;
Digital has “Christmas tree lights”

Resolution and Image Quality

- **Resolution ≠ pixel count!**
  - Some pixels masked as black reference
  - Can interpolate to any pixel count
  - A function of lens, sensor, & processing
- Image Quality (Compression) settings:
  - Raw formats save sensor data as digitized
  - JPEGs are interpolated and compressed; RGB becomes YUV, gamma and DCT applied

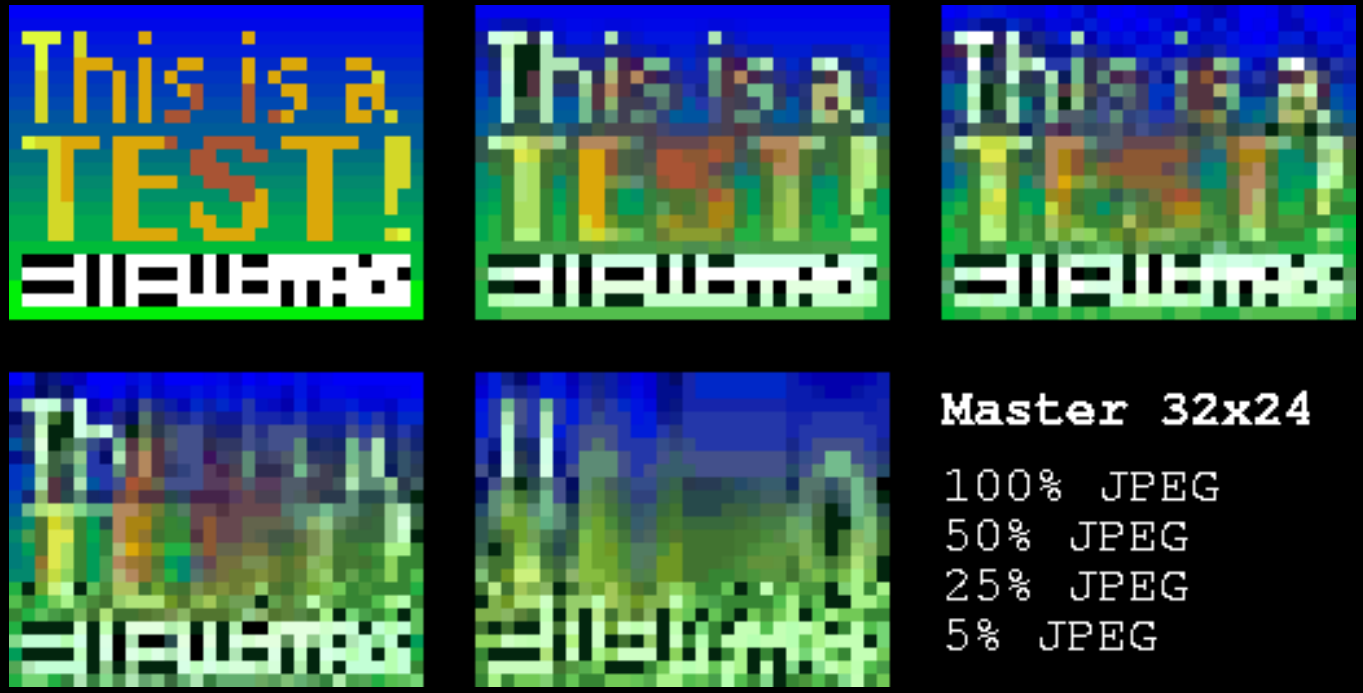
Resolution and Image Quality

61002 vs. 9025 vs. 16309 Bytes
50% @ 256x256 better than 100% @ 128x128

Resolution and Image Quality

196932 vs. 13687 vs. 5735 Bytes
JPEG compression is effective for natural scenes

Resolution and Image Quality



Even 100% JPEG is far from perfect!

Color Balance

- Color reproduction & perception is tricky stuff
  - Use manual white balance where possible
  - Can fix in postprocessing, best from raw

Digital Darkroom Techniques

- Can do some in the camera; preview and options
- Corrections
  - Fix underexposure increases noise
  - Overexposure clips, losing data, so guess?
  - Adjust color, contrast, dodge/burn
  - Fix red eye, remove objects, etc.
- Cropping: 4:3 or 3:2 becomes 7:5, 10:8, etc.

Printing

- Printing technologies (want >100 pixels/inch):
  - Dye Sublimation: high quality, slow, pricey
  - Injet: good quality, high ink+paper costs
  - Laser: faster, cheaper, lower quality
  - Why bother? Just view electronically...
- Many cameras can directly print
- Does print match monitor? **NO!**

You Don’t Print Every Photo: An Important Disneyworld Photo

Where did you park?

Advanced Darkroom Techniques

- Remove, replace, or synthesize elements
  - Patch-based in-painting
  - Sky replacement, deep fakes
- Panorama stitching, superresolution
- Correction of lens/perspective distortions: undo barrel/pincushion, lens tilt, color fringing, etc.
- Special effects (think instagram filters)

Advanced Darkroom Techniques

Panorama of the KAOS Lab, Summer 2002...
13,700x1,920 pixel, i.e., about 25MP...
shot using a 3MP camera!

Computational Photography

- Not about a pretty image on the sensor… but treating the camera as a programmable sensor
- Photo captures information, algorithms transform
  - Depthmaps, 3D, and scene modeling
  - High Dynamic Range (HDR)
  - Synthesis and transformation of images

Computational Photography

Credible repair of Fuji X10 “white orb” defect

Computational Photography

CHDK: Canon Hack Development Kit

Computational Photography

CHDK Lua

Canon Hack Development Kit
Lua scripting reference card

Version 20131022 for `CHDK 1.3.0`

`http://aggregate.org/DIT/CHDK/`

Prof. Hank Dietz
Electrical and Computer Engineering Dept.
University of Kentucky
Lexington, KY 40506-0046
`hankd@engr.uky.edu`

Overview

CHDK, the Canon Hack Development Kit, gives various Canon powerShot cameras new abilities, including the ability to run scripts written in uBASIC or Lua. Recent improvements even allow Lua commands to be exected via USB tethering.

There are many alternative ways to do things in Lua, both functions and constants: 0/1 usually can be `false`/`true`. Some functions listed on a single line to save space.

Focus, IS, & Zoom

mm=`get_focus()`; `set_focus(`*mm*`)`
focus distance in mm when shooting

v=`get_focus_mode()`
0=auto, 1=manual, 3=∞, 4=macro, 5=supermacro

v=`get_focus_ok()`
0=focus not ok, 1=ok *iff* `get_focus_state()`~=0 and `get_shooting()`==1

v=`get_focus_state()`
0=failed, >0=auto success, <0=manual

`set_aflock(`*lock*`)`
lock/unlock autofocus

v=`get_IS_mode()`
image stabilization mode; 0 continuous, 1 shoot only, 2 panning, 3 off

s=`get_zoom()`; `set_zoom(`*s*`)`; `set_zoom_rel(`*s*`)`
zoom position in steps, or +/- relative steps

`set_zoom_speed(`*speed*`)`
set zoom to *speed*% of maximum (typically 5% to 100%)

v=`get_zoom_steps()`
number of zoom steps supported

v=`get_dofinfo()`
depth of field fields: `hyp_valid`, `focus_valid`, `aperture`, `coc`, `focal_length`, `eff_focal_length`, `focus_near`, `far`, `dof`, `hyp_dist`, `min_stack_dist`

Exposure

Exposure parameters can be measured in many different units. APEX (Additive system of Photographic EXposure) uses a log scale in which Ev=Av+Tv=Bv+Sv; Canon/CHDK uses APEX*96 for exposure. Ev is exposure, Av is aperture, Tv is shutter time (-96*log2(seconds)), Bv is luminance, and Sv is ISO sensitivity. Values can be actual `real` (aka `direct`) or rounded `market` values. Functions named `user` are for Manual exposure mode and ones with `id` select by index in table of camera values. Functions use `aperture`*1000; `rel` means +/- offset from current value.

v=`get_av96()`; `set_av96_direct(`*a*`)`

`set_av96(`*a*`)`

v=`aperture_to_av96(`*a*`)`

v=`av96_to_aperture(`*a*`)`

v=`get_bv96()`

v=`get_ev()`; `set_ev(`*a*`)`

v=`get_sv96()`; `set_sv96(`*s*`)`

v=`get_iso_real()`; `set_iso_real(`*a*`)`

v=`get_iso_market()`

v=`get_iso_mode()`; `set_iso_mode(`*a*`)`
market value or 0=auto ISO

v=`iso_to_sv96(`*s*`)`; *v*=`sv96_to_iso(`*s*`)`

v=`iso_real_to_market(`*s*`)`

v=`iso_market_to_real(`*s*`)`

v=`sv96_real_to_market(`*s*`)`

v=`sv96_market_to_real(`*s*`)`

t=`get_tv96()`; `set_tv96_direct(`*t*`)`

`set_tv96(`*t*`)`

v=`get_user_av_id()`; `set_user_av_id(`*a*`)`

v=`get_user_av96()`; `set_user_av96(`*a*`)`

`set_user_av_id_rel(`*a*`)`

`set_user_tv96(`*t*`)`

`set_user_tv_id(`*t*`)`; `set_user_tv_id_rel(`*t*`)`

v=`usec_to_tv96(`*t*`)`; *v*=`tv96_to_usec(`*t*`)`

v=`seconds_to_tv96(`*n*`,`*d*`)`
converts *n*/*d* seconds into tv96 units

v=`get_nd_present()`
have neutral density filter? 0=no, 1=yes, 2=yes+aperture

`set_nd_filter(`*v*`)`
controls neutral density filter: *v*=0 off, 1 in, 2 out

h,*t*=`get_live_histo()`
returns live histogram and total number of pixels

Camera Functions

v=`get_drive_mode()`
0=single shot, 1=continuous, 2,3=self timer

v=`get_flash_mode()`
flash mode: 0=auto, 1=on, 2=off

v=`get_flash_params_count()`
number of flash memory (not strobe) parameters

v=`get_flash_ready()`
flash ready to fire? 0=no, 1=yes

v=`get_meminfo()`
fields: `name`, `chdk_malloc`, `chdk_start`, `chdk_size`, `start_address`, `end_address`, `allocated_size`, `allocated_peak`, `allocated_count`, `total_size`, `free_block_max_size`, `free_block_count`, `free_size`

rec,*vid*,*mode*=`get_mode()`
rec `true` if in record mode, *vid* `true` if in video mode, *mode* is magic mode number

v=`get_movie_status()`
video recorded to SD? 0,1=stopped/paused, 4=recording, 5=stopped but writing to SD card

v=`get_orientation_sensor()`
returns camera orientation in degrees

str,*num*=`get_parameter_data(`*id*`)`
reads flash memory parameter *id*

v=`get_prop(`*p*`)`; *v*=`set_prop(`*p*,*v*`)`
access PropertyCase value

v=`get_prop_str(`*p*`)`; *s*=`set_prop_str(`*p*,*v*`)`
access PropertyCase string value

v=`get_propset()`
identifies PropertyCase set used by this camera

v=`get_shooting()`
ready to shoot? (half press, focus, and exposure set)

v=`get_temperature(`*w*`)`
reads temperture of 0=optics, 1=sensor, 2=battery

v=`get_vbatt()`
read battery voltage in mV

v=`get_video_button()`
does camera have a video button? 0=no, 1=yes

v=`is_capture_mode_valid(`*n*`)`
true if *n* is a valid mode number

v=`set_capture_mode(`*n*`)`
sets mode and returns true if in record mode

v=`set_capture_mode_canon(`*n*`)`
sets mode by PropertyCase and returns true if camera is in record mode

`set_led(`*a*,*b*,*c*`)`
a is LED number; *b*=0 off or 1 on; *c* is brightness 0-200

`set_movie_status(`*v*`)`
1=pause recording video, 2=resume recording, 3=stop recording

`set_record(`*v*`)`
0 (or false) sets play mode, 1 (or true) sets record

`shut_down()`
like `post_levent_to_ui('PressPowerButton')`

Buttons

Buttons are camera dependent, although all have `"shoot_half"` and `"shoot_full"`.

`click(`*button*`)`
simulate press, then release, of button *b*

v=`is_key(`*button*`)`; *v*=`is_pressed(`*button*`)`
1 if *button* was; is being pressed

`press(`*button*`)`; `release(`*button*`)`

`shoot()`

`wait_click(`*{t}*`)`
wait up to *t*/1000s for any key to be clicked

`wheel_left()`; `wheel_right()`
simulate wheel move one click ccw; cw

`set_exit_key(`*b*`)`
set *b* as the key to terminate this script

SD Card Functions

v=`get_disk_size()`
size of SD card in KB (1024B) units

v=`get_exp_count()`
number of shots in a session

v=`get_image_dir()`
directory where most recent exposure was written

fe=`file_browser(`*path*`)`
lets user select a file

v=`get_free_disk_space()`
space remaining on SD card in KB (1024B) units

v=`get_jpg_count()`
number of JPG shots that would fit on SD card

part=`get_partitionInfo()`
fields: `count`, `active`, `type`, `size`

`set_file_attributes(`*fe*,*a*`)`
set attributes of *fe* to bits in *a*: 0x1=read only, 0x2=hidden, 0x20=archive

`swap_partition(`*n*`)`
make partition *n* active

Time & Scheduling

v=`autostarted()`
return 1 (true) is script was autostarted

v=`get_autostart()`; `set_autostart(`*v*`)`
autostart can be 0=off, 1=on, 2=once

v=`get_tick_count()`
clock time in 1/1000s units

v=`get_time(`*unit*`)`; *v*=`get_day_seconds()`
time specified by *unit* string: `Year`, `Month`, `Day`, `hour`, `minute`, `second`; or simply seconds since midnight

oc,*ms*=`set_yield(`*c*,*ms*`)`
set maximum number of Lua VM instructions to contiguously execute as *c*100 and maximum time as *ms*; old values are returned

`sleep(`*time*`)`
Sleep for time in 1/1000s units

Display & Text Console

`set_backlight(`*v*`)`
LCD backlight on/off

i=`get_draw_title_line()`;`set_draw_title_line(`*i*`)`
CHDK <ALT> line on LCD on/off

`cls()`; `console_redraw()`
clear/redraw mini-console screen

`print(`...`)`
write args to mini-console

`print_screen(`*nnnn*`)`
if *nnnn*=0, disables echo to log file; >0 logs to new file `LOG_nnnn.TXT`; <0 appends to log file

`set_console_autoredraw(`*n*`)`
n=1 enables auto update of log file and LCD; 0 disables; -1 updates log file only

`set_console_layout(`*x1*,*y1*,*x2*,*y2*`)`
position and size in characters; 0,0,45,14 is full screen

LCD Graphics

Drawn on LCD, but overwritten by any update. Colors are non-portable 0-255 Canon palette or portable: 256 (`transparent`), 257 (`black`), 258 (`white`), 259 (`red`), 262 (`green`), 265 (`blue`). Edge thickness also can be set.

`draw_clear()`

`draw_ellipse(`*x*,*y*,*a*,*b*,*c*`)`

`draw_ellipse_filled(`*x*,*y*,*a*,*b*,*c*`)`

`draw_line(`*x1*,*y1*,*x2*,*y2*,*c*`)`

`draw_pixel(`*x*,*y*,*c*`)`

`draw_rect(`*x1*,*y1*,*x2*,*y2*,*c*,*thick*`)`

`draw_rect_filled(`*x1*,*y1*,*x2*,*y2*,*c* *ll*,*c*,*thick*`)`

`draw_string(`*x*,*y*,*text*,*cf*,*cb*`)`

`set_file_attributes(`*fe*,*a*`)`
set attributes of *fe* to bits in *a*: 0x1=read only, 0x2=hidden, 0x20=archive

`textbox(`*title*,*prompt*,*def*,*maxlen*`)`
gets a string from user input

Raw

v=`get_raw()`; `set_raw(`*v*`)`
enable/disable saving raw images

v=`get_raw_count()`
number of raw shots that would fit on SD card

v=`get_raw_nr()`; `set_raw_nr(`*v*`)`
noise reduction enabled/disabled

`raw_merge_start(`*op*`)`
start raw merging; *op* can be 0 (sum) or 1 (average)

`raw_merge_add(`*fe*`)`
adds raw *fe* to the merge

`raw_merge_end()`
completes merge; result is `SND_XXXX.CRW`, where *XXXX* is `get_exp_count()` % 10000

`set_raw_develop(`*fe*`)`
next shot develops raw *fe* into JPEG

CHDK Functionality

`enter_alt()`; `exit_alt()`
enter/exit CHDK <ALT> mode

v=`get_buildinfo()`
fields: `platform`, `platformid`, `platsub`, `version`, `os`, `build_number`, `build_revision`, `build_date`, `build_time`

i1[,*i2*][,*s*][,*t*]=`get_config_value(`*Con gId*[,*def*]`)`
get specified CHDK configuration value

v=`get_histo_range(`*lo*,*hi*`)`
percentage raw buffer pixels in [*lo*, *hi*]

`set_config_value(`*Con gId*[,*i1*][,*i2*][,*s1*][,*t*]`)`
set specified CHDK configuration value

`shot_histo_enable(`*v*`)`
enable/disable computing shot histograms

Programming

v=`bitand(`*a*,*b*`)`
bitwise and; also `bitor`, `bitxor`, `bitshl` (<<), `bitshri` (int >>), `bitshru` (unsigned >>)

v=`bitnot(`*a*`)`

v=`peek(`*addr*[,*size*]`)`; *s*=`poke(`*addr*,*v*[,*size*]`)`
load/store memory[*addr*]; size is 1/2/4, default 4, for char/short/int

v=`call_func_ptr(`*fptr*,...`)`
calls compiled C function at ARM address *fptr*, returns `R0`

Motion Detection

v=`md_motion_detect(`...`)`
number of zones in which motion was detected; many arguments control detection

v=`md_get_cell_diff(`*x*,*y*`)`
returns unsigned [0,255] difference in last two readings of cell *x*,*y*

v=`md_get_cell_val(`*x*,*y*`)`
returns unsigned [0,255] value of cell *x*,*y* (for Y, U, V, R, G, or B channel specified)

`md_af_on_time(`*d*,*t*`)`
show motion detected by autofocus assist lamp; delay *d**10ms before on; *t**10ms before off; 0,0 disables

Tone Curves

Only for cameras using 10-bit raws. There are 5 states, 0-4: no curve, custom file, +1 Ev, +2 Ev, and auto dynamic range enhancement.

v=`get_curve_state()`; `set_curve_state(`*v*`)`
get/set tone curve state

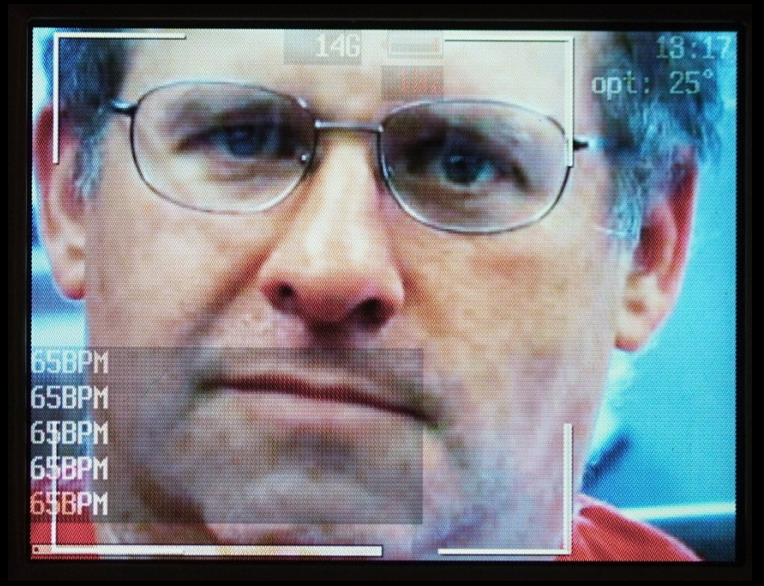
fe=`get_curve_file()`; `set_curve_file(`*fe*`)`
get/set currently loaded tone curve

CHDK Lua (also supports BASIC, native C)

Computational Photography

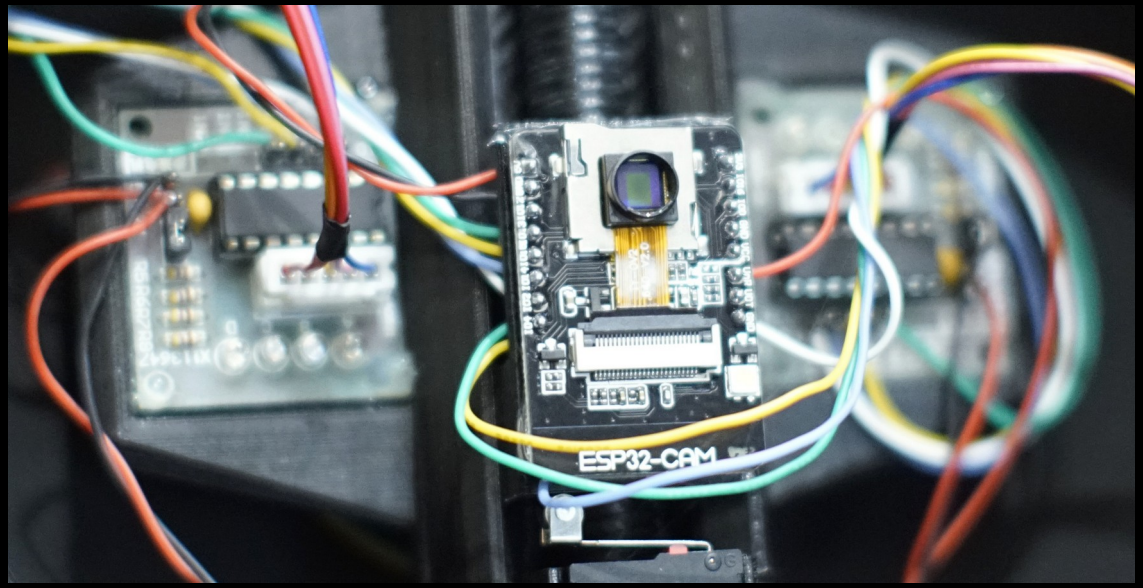

```
--[[
@title So Far Away
-]]

function mypoll(t)
     local v=0
     local feet=0
     repeat
          sleep(t)
          press("shoot_half")
          repeat
               sleep(50)
          until get_shooting() == true
          v=get_focus()
          v=10*v
          v=v+127
          v=v/254
          feet=v/12
          v=v%12
          print(feet .. " feet " .. v .. " inches")
          sleep(t)
          release("shoot_half")
     until (false)
end

mypoll(1000)
```

Computational Photography



Photoplethysmography using a Canon PowerShot

Computational Photography



Arduino-compatible IoT (Internet Of Things) parts:

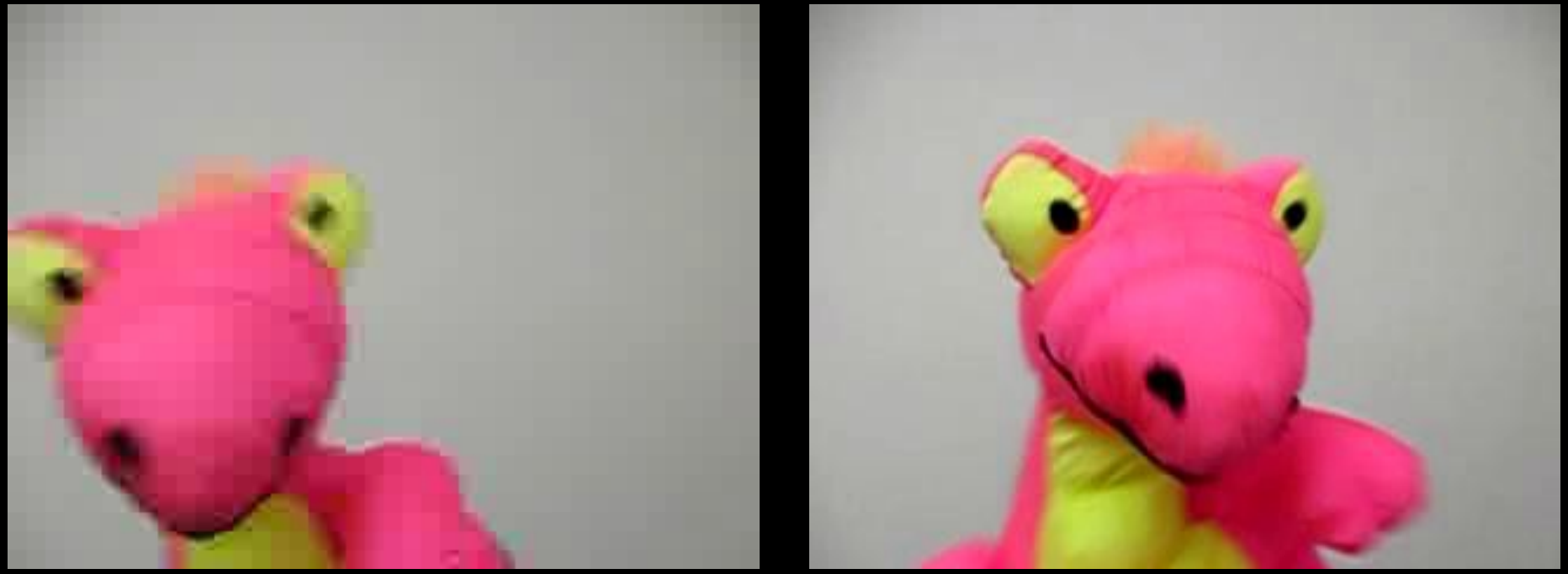
$7 **ESP32-CAM**: 2MP, dual core, TF, WiFi, etc.

Computational Photography

This day and age we're living in
gives cause for apprehension
With speed and new invention
and things like fourth dimension.
Yet we get a trifle weary
with Mr. Einstein's theory. ...
The fundamental things apply
as time goes by.

TDCI: Time Domain Continuous Imaging

Computational Photography



Original vs. TDCI rendering of 240FPS video

Computational Photography

TDCI renderings at 24FPS and 100FPS

Conclusion

- Digital cameras are now cheap and very good
- They can be treated as:
  - A medium for artistic expression
  - A visual record, available even in real time
  - Programmable sensors and computers

http://aggregate.org/DIT/