Introduction to the ASPEN Eagle Eye Model

Michael Trowbridge, 1 May 2018

Jet Propulsion Laboratory
California Institute of Technology

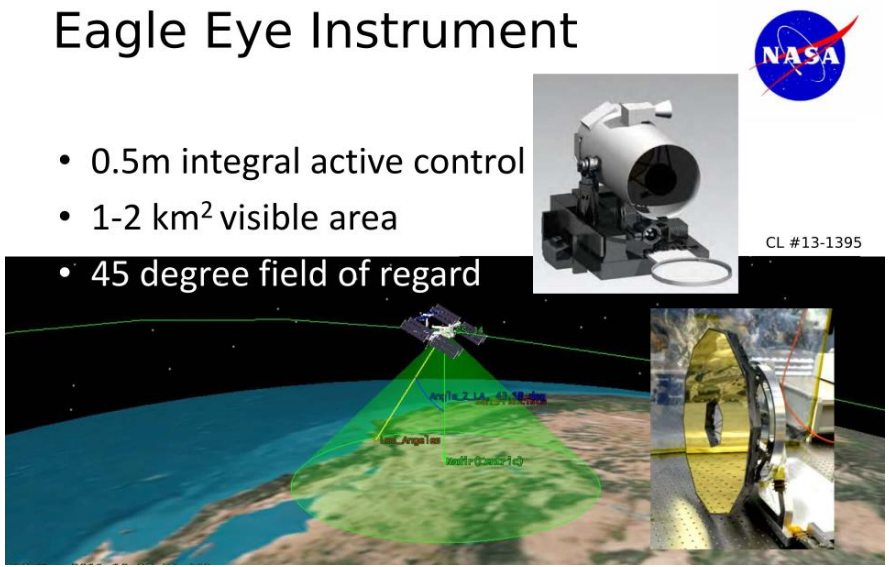
Copyright 2018, by the California Institute of Technology. ALL RIGHTS RESERVED. United States Government Sponsorship acknowledged.

What is the Eagle Eye ASPEN Model?

- Specialized ASPEN C++ scheduling function library for observations with 2D framing instruments
- Runs loosely under ASPEN
- Ephemeris: JPL NAIF SPICE toolkit
- Target geometry based on Compressed Large Activity Scheduler Planner (CLASP)

What are we modeling?

- Fixed bus, Gimballed Instrument:
  - Proposed Eagle Eye: bus is the ISS and outside of our control. Instrument is gimballed (2-axis) relative to the spacecraft.
- Steerable bus, Body-fixed Instrument:
  - Proposed GAZE/TeMBLOR
  - GOLIAT CubeSat
  - ALL-STAR 3U CubeSat
- Steerable bus, Gimballed Instrument
  - Proposed Mission to Understand Ice Retreat (MUIR) Free-Flyer concept



How does Eagle Eye contribute to a mission?

Scheduling as simulation for pre-phase A Mission Design Questions

- How much of the reference science campaign will a proposed mission achieve?
- What does the instrument/spacecraft pointing profile look like for a typical mission period?
  - Where are we pointing the instrument/spacecraft?
  - What angular rates must our attitude actuators handle?
  - Can you make a cool video for my presentation?
- Are our science products qualitatively viable?
- How much data should we budget for science?
- Is our mission more successful if we host our payload on another spacecraft (ISS) or procure/design our own bus?

Inputs, Output Products

Products used by mission design

Inputs

- Orbit/trajectory description
- Instrument specifications
- Instrument gimbal/spacecraft agility
- Planning horizon
- Data recorder configuration
- Ground station configuration
- Science targets KML file

Eagle Eye

Outputs

- Time-tagged, Comma-Separated Values report (vector report)
  - Position, attitude, rates
  - Pointing target
  - Memory/downlinks
- SPICE Attitude Kernels
- Google Earth KML files
- Cosmographia directory

Tutorial outline

- For a hypothetical mission, we will:
  - Generate a SPICE kernel in STK
  - Create an Eagle Eye-derived ASPEN model
  - Answer the typical Mission Design analysis questions
  - Examine some of our output products
- Live demonstration anywhere there is a screenshot or Google Earth image.

Prerequisites for this tutorial

- Linux or OSX
- Many AI group git repositories checked out & compiled

  git clone git@github.jpl.nasa.gov:ASPEN/aspen.git

  git clone git@github.jpl.nasa.gov:CLASP/clasp.git

  git clone git@github.jpl.nasa.gov:AIG/cspice.git

  git clone git@github.jpl.nasa.gov:ASPEN/aspen-eagle-eye-models.git

  git clone git@github.jpl.nasa.gov:ASPEN/star-data.git
- Cosmographia
- Systems ToolKit (STK - installed via Parallels/virtual machine)

The Hypothetical Mission

Mission Description

Sensorweb extension: monitor volcanos from space. Look for deformations that may indicate a future eruption or new vent.

Baseline Orbit

Sun-synchronous, ascending node at Solar noon, 615 km circular orbit (similar to WorldView-4).

Baseline Spacecraft Design

- Steerable bus, body-fixed instrument.
- 16 Gbyte data recorder, 160 megabits/sec I/O rate
- Instrument
  - 1 degree x 9/16 degree instrument field of view.
  - 1920x1080 pixels @ 10 bits/pixel + 65535 bytes metadata
  - No compression
  - Up to 17 frames/sec
- Ground station: Goldstone
  - 35.4267 deg N, 116.89 deg W, 256 kbits/sec downlink
- Constraint: instrument boresight >= 90 degrees away from the Sun.

Setting up the ASPEN model

Create a model directory

- Files:
  - Makefile
  - model.mdl: copy from $EAGLE_EYE_ROOT
  - Optional:
    - parameter-functions.{cc,h}, scheduling-functions.{cc,h}, userfunc.{cc,h}
- Directories
  - kernels/: contains every SPICE kernel used for the spacecraft/CSPICE initialization
  - Start by copying $EAGLE_EYE_ROOT/kernels
- Location: peer to eagle_eye

```
# Common make variables for the appropriate platform and compiler.
include $(ASPEN_ROOT)/config/make.cfg
# Make variables for model functions (dependencies, heuristics, etc.)
include $(ASPEN_ROOT)/config/model-make.cfg

# The ASPEN types and instances
MDL_FILES      = horizon.mdl model.mdl spacecraft.mdl instrument.mdl mission.mdl
INI_FILES      = model.ini
HTAB_FILE      = $(ASPEN_ROOT)$(DIR_SEP)models$(DIR_SEP)heuristics.tab

# The external libraries (Eagle Eye) for user defined scheduling functions
EXT_LIB_DIR = $(EAGLE_EYE_ROOT)
EXT_PFUNC_SHLIB = $(EXT_LIB_DIR)$(DIR_SEP)libeagleeyepfunc$(SHLIB_EXT)
EXT_SFUNC_SHLIB = $(EXT_LIB_DIR)$(DIR_SEP)libeagleeyesfunc$(SHLIB_EXT)

LD_LIBRARY_PATH_RUN=$(LD_LIBRARY_PATH):$(EXT_LIB_DIR)
# DYLD_FALLBACK_LIBRARY_PATH is Mac specific but doesn't break other OSs
LD_LIBRARY_PATH_VAR=LD_LIBRARY_PATH=$(LD_LIBRARY_PATH_RUN)
DYLD_FALLBACK_LIBRARY_PATH=$(LD_LIBRARY_PAT    H_RUN)

# Command make rules
include $(ASPEN_ROOT)/config/make.rls
# Make rules for model functions and libraries
include $(ASPEN_ROOT)/config/model-make.rls

schedule_targets: precomputed_volcanos.kml
	-make -C $(EAGLE_EYE_ROOT)
	-# Cannot run "make run" on OSX because the OS blocks
	-# DYLD_FALLBACK_LIBRARY_PATH environment variable through make
	-$(LD_LIBRARY_PATH_VAR) aspen -b -s 12345 -m $(MDL_FILES) -i $(INI_FILES) -f schedule_targets_from_files -f vector_report –f
vector_report_kml -f output_memory_csv 1 -f outputCosmographia ./ hypothetical -h $(HTAB_FILE) -F libeagleeyepfunc.dylib -UF
libeagleeyesfunc.dylib

precomputed_volcanos.kml: volcanos.kml
	$(LD_LIBRARY_PATH_VAR) aspen -b -s 12345 -m $(MDL_FILES) -i $(INI_FILES) -f precompute_visibilities volcanos.kml
precomputed_volcanos.kml -h $(HTAB_FILE) -F libeagleeyepfunc.dylib -UF libeagleeyesfunc.dylib

solutionclean:
	$(RM) -f vector_movie.*
	$(RM) -f vector_report.*
	$(RM) -f gps_ground_track.kml
	$(RM) -f aspen.log debug.kml
```

Setting up the ASPEN model

Ways to generate a trajectory

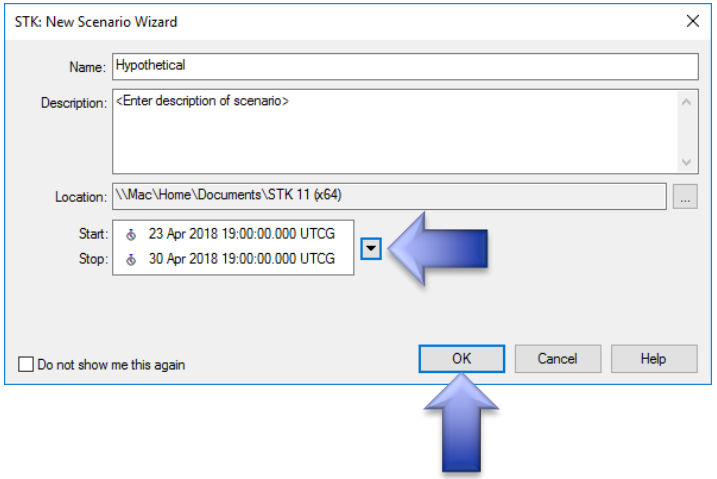
- Ask Mission Design/Navigation to do it.
  - Two-Line Element (TLE) set: Ask for a TLE and put it in a text SPICE kernel. <24-hour planning horizon (SPICE restriction).
  - BSP: have MD/Nav propagate the trajectory in MONTE and export a binary SPICE kernel (BSP).
- Propagate from STK (and export BSP).
- Propagate from SOAP (and export BSP).

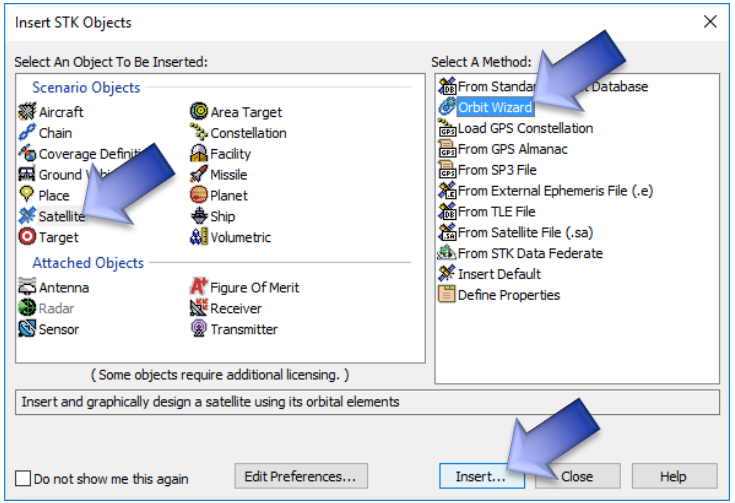
I prefer this way

How we'll do it

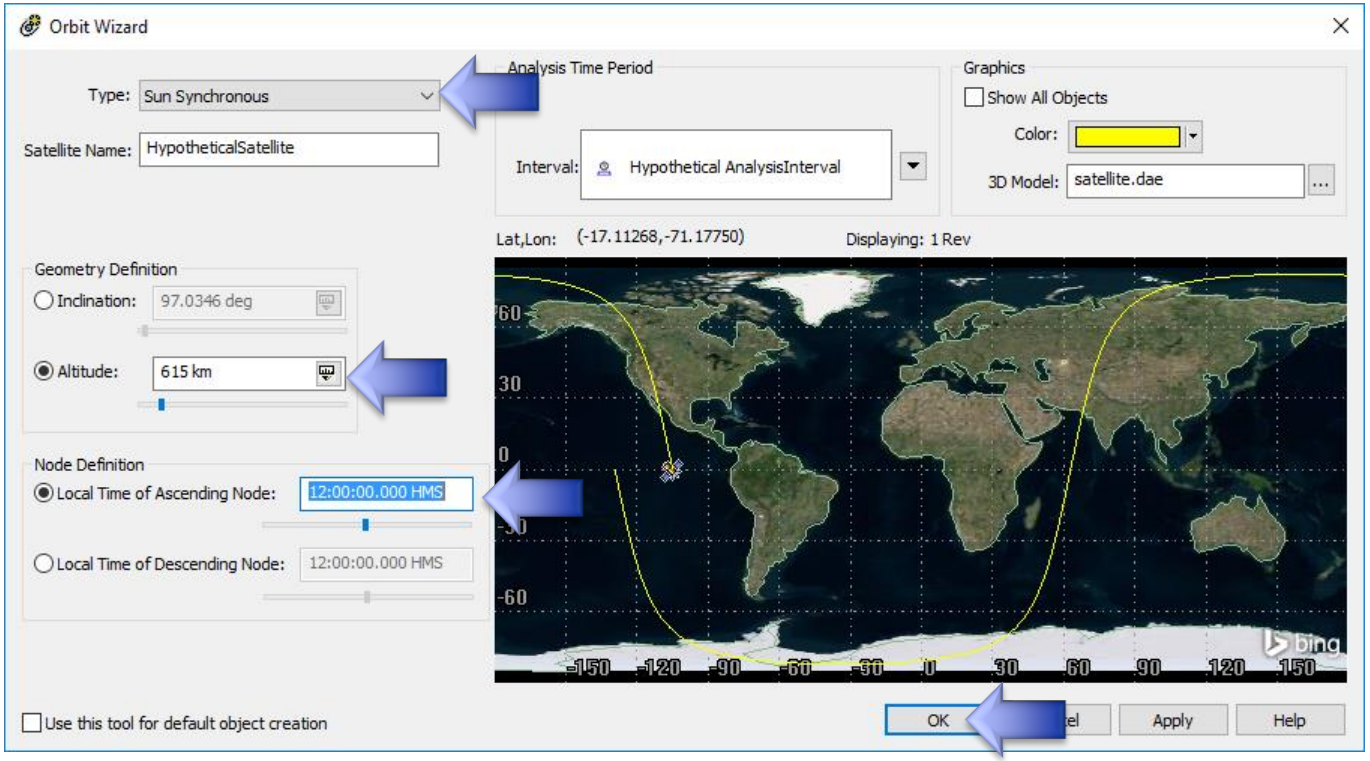
Propagating and exporting a trajectory from STK

Create the STK Scenario

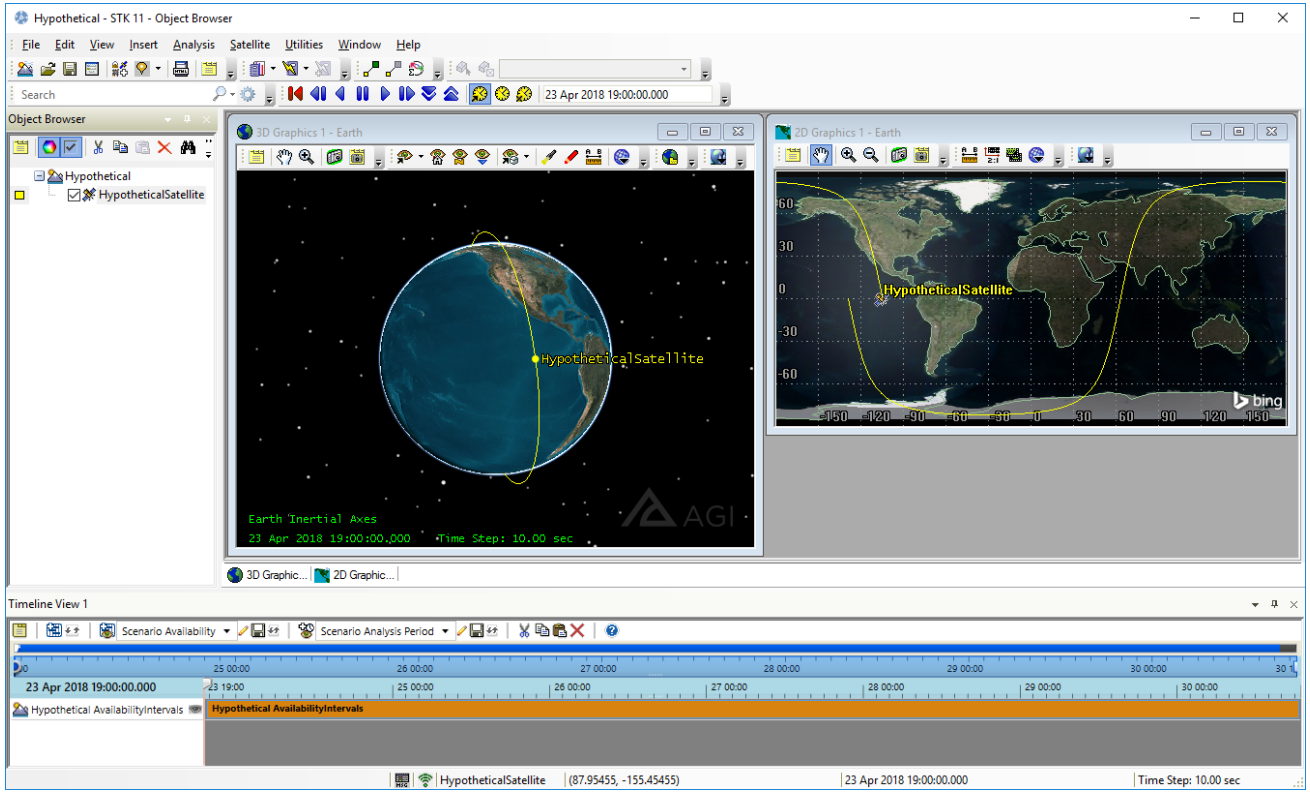




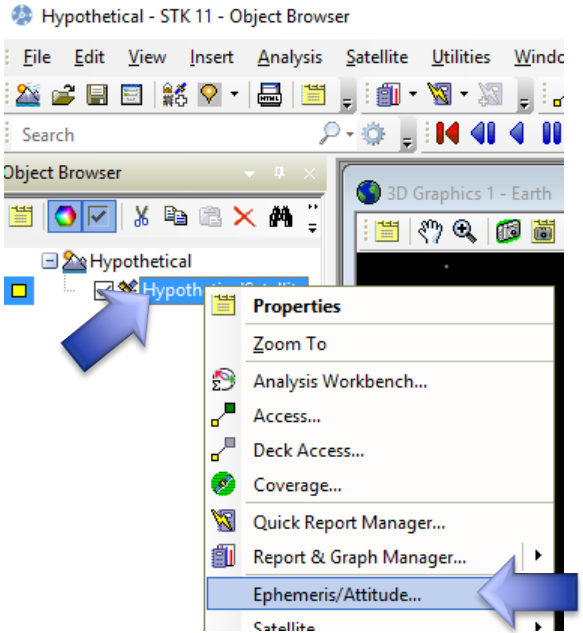
Make the orbit using the wizard

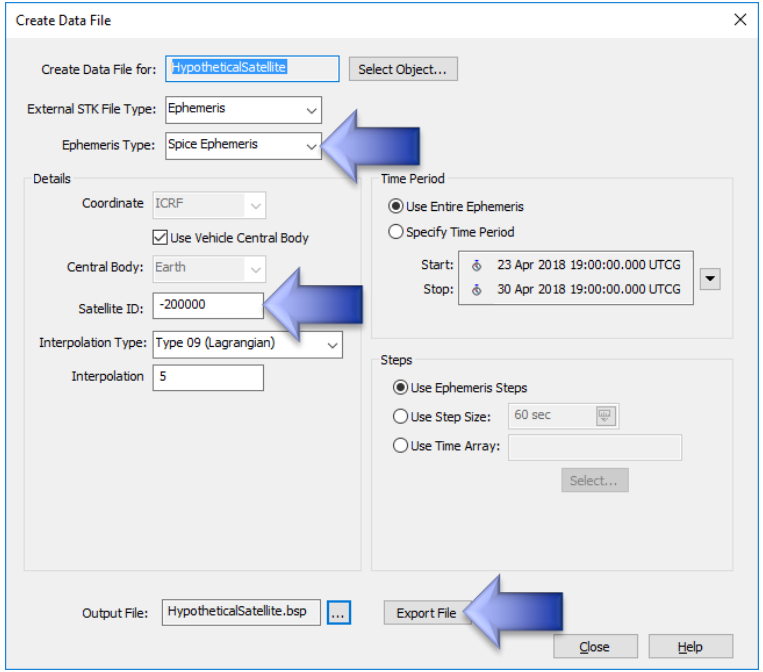


Visually check the orbit in STK



Export the Binary SPICE Kernel from STK





Setting up the ASPEN model

Instrument Description

- Telescope FOV: range that the gimbal instrument can be steered
  - Specified in (w,h) angle (rad) around a home position
- Instrument FOV: instantaneous field of view of the instrument
  - Specified in (w,h) angle (rad) around instrument's current boresight target
- Instrument agility (t_slew, t_settle) = f(angle)
  - Lookup table
  - Best-case, worst-case bounds only
  - Anything else: write a custom functor in a user function library and bind it.
- Size of an image
- Frame rate of the instrument

```
$ cat instrument.mdl
// (1920x1080x10+ 65535*8 bits/image) /1e6 bits per megabit
parameter double image_memory_size {domain = 21.2608;}

parameter double compression_factor {domain = 1;}

parameter double frame_rate { domain=[0.0, 17.0]; }
parameter double fps { domain=17.0; }
parameter double instrument_fovs_per_second { domain = 17.0; }

parameter double telescope_fov_degrees_width { domain = (1.0); }
parameter double telescope_fov_degrees_length { domain = (0.5625); }
parameter double instrument_fov_degrees_width { domain = (1.0); }
parameter double instrument_fov_degrees_length { domain = (0.5625); }
```

Only applies to gimballed instruments

Setting up the ASPEN model

Spacecraft Description

- Bus agility (t_slew, t_settle) = f(t, attitude_start, attitude_end)
  - Lookup table
  - Other models possible: acceleration envelopes, custom code
- Trajectory (TLE or SPICE kernel)
- NAIF Spacecraft ID in the SPICE kernel
- Data recorder specifications
  - Maximum I/O rate
  - Capacity
- Spacecraft pointing constraints

```
$ cat spacecraft.mdl
parameter double settle_step_degrees {default = 1.25;}
parameter vector<double,double,double> settle_durations_vector {default=<0.0,0.05,0.1>;}

parameter double slew_step_degrees { default = 180.0; }
parameter vector<double,double> slew_durations_vector { default = <0.0,120.0>; }

parameter string spacecraft_1_id        {domain = ("-200000");}
parameter string kernel_1_filename { domain = ("kernels/HypotheticalSatellite.bsp"); }

// 16 gigabytes * 8 bits per byte * 1000 megabytes per gigabyte
parameter double onboard_memory_size {domain = 12800.0;}

// 160 megabits/sec
parameter double max_raw_image_data_rate {domain = 160;}
```

Setting up the ASPEN model

Spacecraft Description: Spacecraft Pointing Constraints (spacecraft.mdl)

```
activity sc_1
{
  // Define spacecraft properties here
  int spice_id = -200000;
  bool is_instrument_gimballed = false;

  // Define spacecraft geometric constraints here
  decompositions = boresight_sun_angle_deg_illuminated,
                   boresight_sun_angle_deg_penumbra;
}
```

```
// Don't point the camera near the sun.
activity boresight_sun_angle_deg_illuminated
{
 // boresight vector in sc frame
  vector<double,double,double> constraintVector = <0.0,0.0,-1.0>;
  int targetSpiceID = 10; // Sun
  real bounds = [90.0,180.0]; // angle bounds in degrees
}

// Don't point the camera near the Sun.
activity boresight_sun_angle_deg_penumbra
{
  // boresight vector in sc frame
  vector<double,double,double> constraintVector = <0.0,0.0,-1.0>;
  int targetSpiceID = 10; // Sun

  real bounds = [90.0,180.0]; // angle bounds in degrees
}
```

Setting up the ASPEN model

Communications Description

- Ground station information
  - Location (lat, lon, alt)
  - Downlink data rate
  - Elevation angle mask
- Assumptions
  - The proposed mission has 100% access to a ground site any time it is in view.
  - The observer can communicate and observe at the same time

```
$ cat mission.mdl
…
parameter double min_station_elevation {domain = 7.777;}
parameter double comm_rate {domain = 0.256;} // 256 kb/s = 0.256 mbps
parameter double comm_max_angle_off_boresight {domain = 32;}
parameter int ground_site_count {domain = 1;}
parameter string comms {
  domain = (
    "goldstone"
  );
}
parameter double goldstone_lat {domain = 35;}
parameter double goldstone_lon {domain = -116;}
parameter double goldstone_alt {domain = 0;}
parameter string goldstone_style {domain = ("aqua");}
state_variable sc_1_goldstone_view {domain = views;}
activity sc_1_goldstone_in_view {
  reservations=
    sc_1_goldstone_view change_to "in view",
    sc_1_goldstone_view change_to "not in view" at_end,
    sc_1_ground_site_in_view_count use 1;
  no_permissions=("move","add","delete","lift");
}
```

Setting up the ASPEN model

More mission.mdl configuration

General configuration

```
$ cat mission.mdl
parameter int time_step {domain = 10;}

parameter int spacecraft_count { domain = 1;}

// This is the high precision Earth orientation data taking
// Precession, Nutation, Nutation corrections, UT1-TAI, and Polar motion
// into account.  Falls back to low precision prediction model
// if your horizon goes outside of the frame's coverage time period.
parameter string body_frame { domain = ("ITRF93"); }

parameter int swo_iterations {domain = 5;}
parameter int swo_priority_delta {domain = 1;} //1 is the default

// Schedule time clustering only applies to gimballed instruments
parameter bool schedule_time_cluster { domain = false; }

// The science campaign files we want to schedule
parameter string KML_FILE_LOCATIONS { domain = ("precomputed_volcanos.kml"); }
…
```

Area-related

```
$ cat mission.mdl
…

// Minimum overlap between two adjacent tiles for mosaicking
parameter double stitching_percent {domain = 10.0;}

// Configuration parameter for an area mosaicking algorithm
parameter string ONLINE_FRONTIER_REPAIR_ROW_ALIGNMENT {
   domain=("CROSS-TRACK","ALONG-TRACK");
   default="CROSS-TRACK";
}

// Configuration for computing how much planning space margin we shoul
// apply to make it more likely that an initial plan complies with
// stitching_percent
parameter string tfov_margin_strategy {
  domain=("MARGIN_SMALLEST_EDGE","MARGIN_BISECTION","MARGIN_MOST_CONSERVATIVE","MARGIN_LE
AST_CONSERVATIVE");
  default="MARGIN_SMALLEST_EDGE";
}
…
```

Setting up the ASPEN model

Configuring the planning horizon and timescale

- Future versions will require time_scale=nanosecond
- Recommend starting with a short horizon_duration to test your model setup initially, then expand it for real analysis runs

```
$ cat horizon.mdl
model Hypothetical {
  time_scale = second;
  time_zone = utc;
  time_format = tee;
  horizon_start = 2018-04-23/19:00:01;
  horizon_duration = 6d;
}
```

Setting up the ASPEN model

Science Description: CLASP Observation Request KML (CORK)

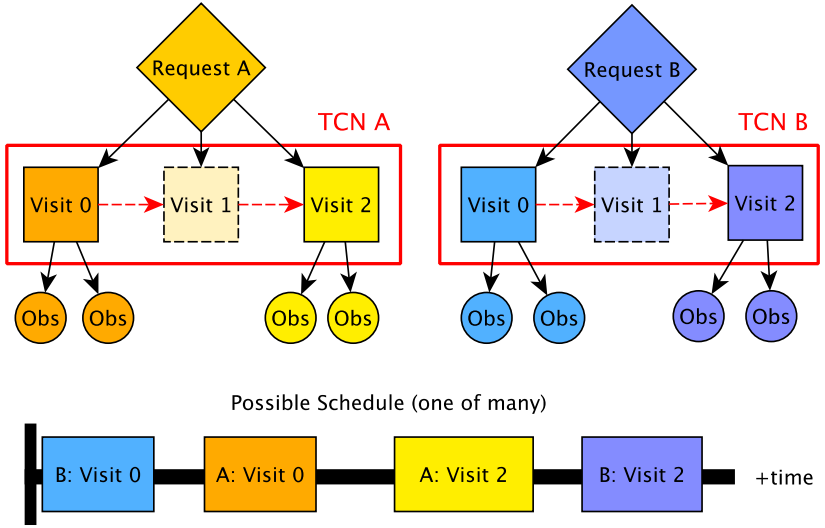
- Targets go in <Placemark> KML elements.
  - Geography
    - Target-fixed: <Point> or <Polygon> supported
    - Time-varying: celestial objects supported if a SPICE kernel is provided in the observation_request CLASP element.
  - CLASP campaign{ observation_request{}; } element inside Placemark/Description element.

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2" xmlns:gx="http://www.google.com/kml/ext/2.2"
 xmlns:kml="http://www.opengis.net/kml/2.2" xmlns:atom="http://www.w3.org/2005/Atom">
 <Folder>
  <name>Africa Volcano campaign</name>
  <Folder>
   <name>Africa (northern)</name>
   <Placemark>
    <name>Todra Volcanic Field</name>
    <description>campaign{
     observation_request {
     id=2;
     priority=100.0;
     frame_duration_sec = 1.0;
     frame_count = 1;
     target_elevation_angle_deg = (35.0,90.0);
     solar_elevation_angle_deg = (45.0,90.0);
     duration_sec = 518390.0;
     revisit_time_sec = (82800.0,90000.0);
     };
     }
    </description>
...
    <Point>
     <coordinates>8.5,17.68,0</coordinates>
    </Point>
   </Placemark>
...
```

Together, these mean 1x daily,+-1 hour

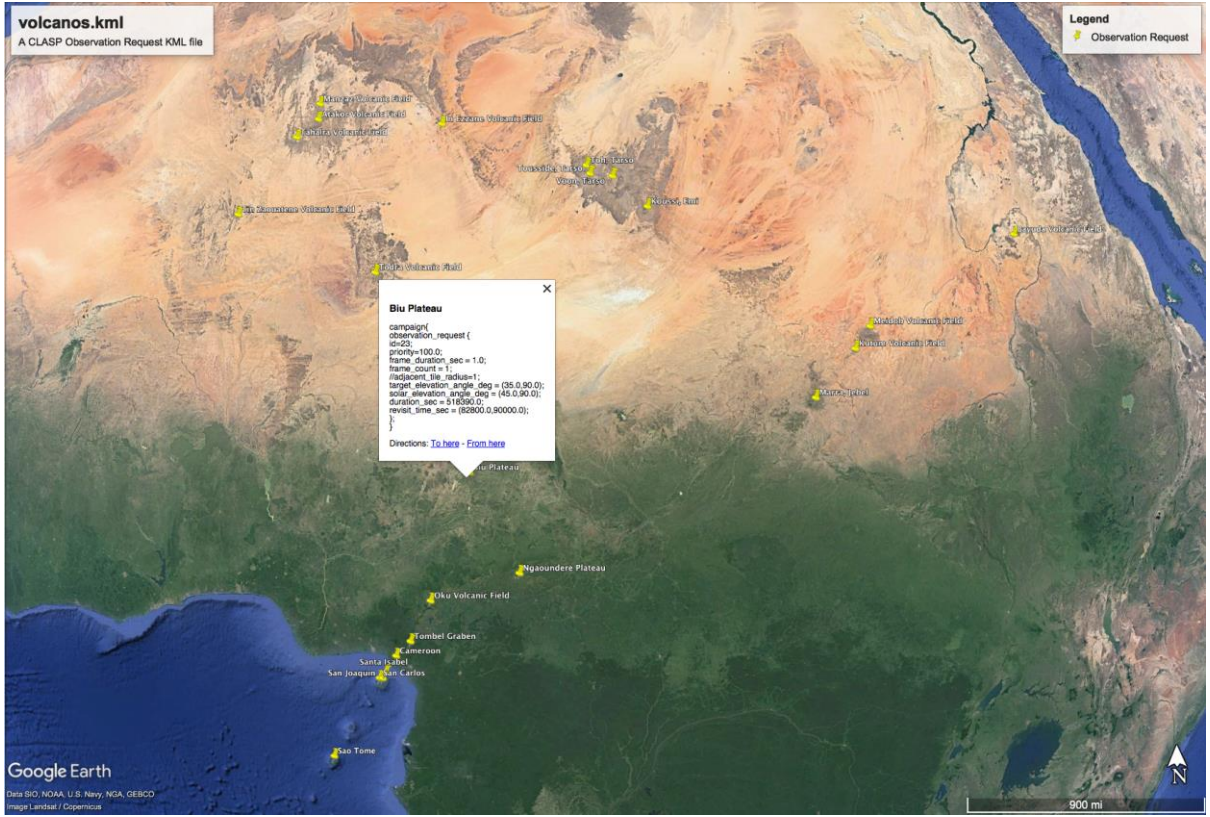
Revisits in Eagle Eye

- Visits form a degenerate temporal constraint network
  - 0 or more visits may be detailed
  - Precedence is mandatory
  - Delay between revisits has a lower and upper bound
  - Revisit tolerances propagate from the most recent detailed visit, expanding through abstract visits
- A Visit contains 1 or more consecutive observations (frame_count in CORK file).
- Visits from one request may be interleaved with visits from another request
- 100% satisfaction: all visits detailed. 0%: all visits abstract.



Lewellen, G.; Davies, C.; Byon, A.; Knight, R.; Shao, E.; Tran, D.; and Trowbridge, M. 2017. A Hybrid Traveling Salesman Problem - Squeaky Wheel Optimization Planner for Earth Observational Scheduling. In Chien, S., and Augenstein, S., eds., *Proceedings of the 10th International Workshop on Planning and Scheduling for Space (IWPSS)*, 62–72.

The target CORK file in Google Earth



Running the model

- At the start of your session:

  cd aspen

  source setup.sh

  cd ../aspen-eagle-eye-models/models/eagle_eye

  source setup.sh

  cd ../hypothetical
- To run it:
  - ~~make run~~ broken on OSX! (OS prevents child process from receiving DYLD_FALLBACK_LIBRARY_PATH)
  - make schedule_targets: runs ASPEN directly
    - No aspen.log file

Answering Mission Design Questions

How much of the mission do we satisfy?

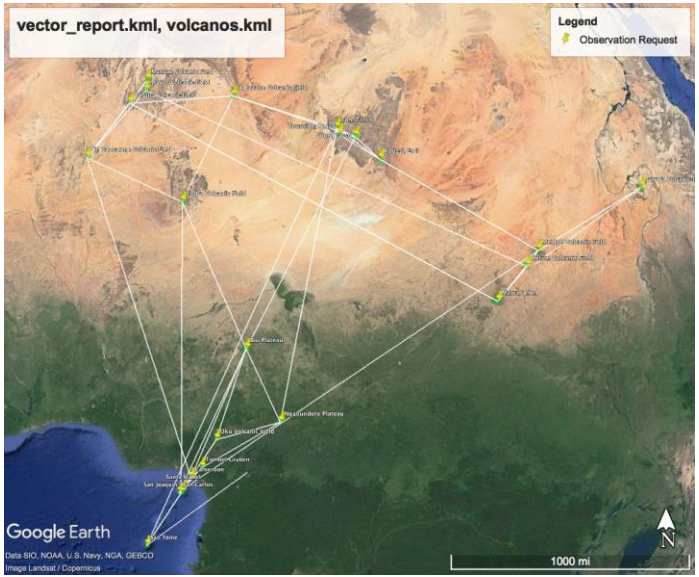
satisfaction.csv

ID	Fraction Complete (0 to 1)
1	0.6
2	0.4
3	0.6
4	0.4
5	0.6
6	0.6
7	0.6
8	0.8
9	0.6
10	0.6
11	0.8
12	0.6
13	0.6
14	0.8
15	0.4
16	0.4
17	0.6
18	0.6
19	0.6
20	0.4
21	0.4
22	0.4
23	0.6

Best: 4 of 5 visits scheduled

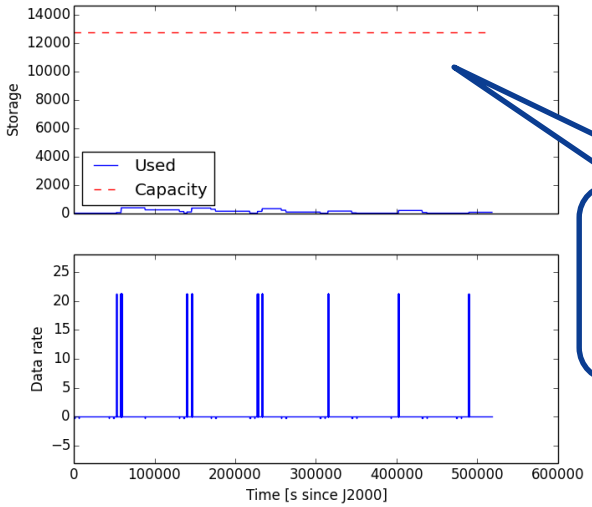
Worst: 2 of 5 visits scheduled

Python plotting scripts for histograms and heat maps are helpful here



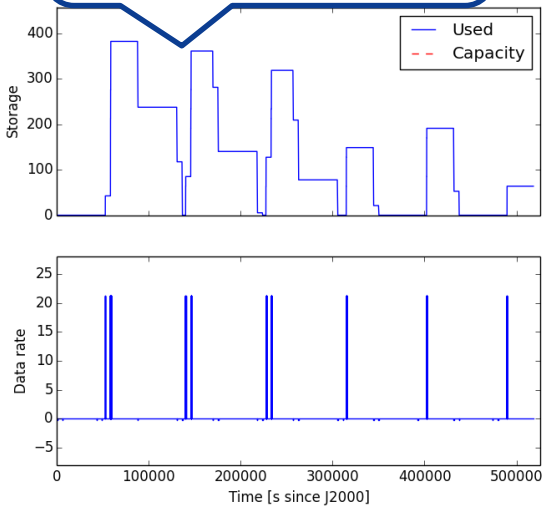
Answering Mission Design Questions

How much data should we budget for science?



Perhaps we don't need so much?

Downlink is the next bottleneck if we add more targets



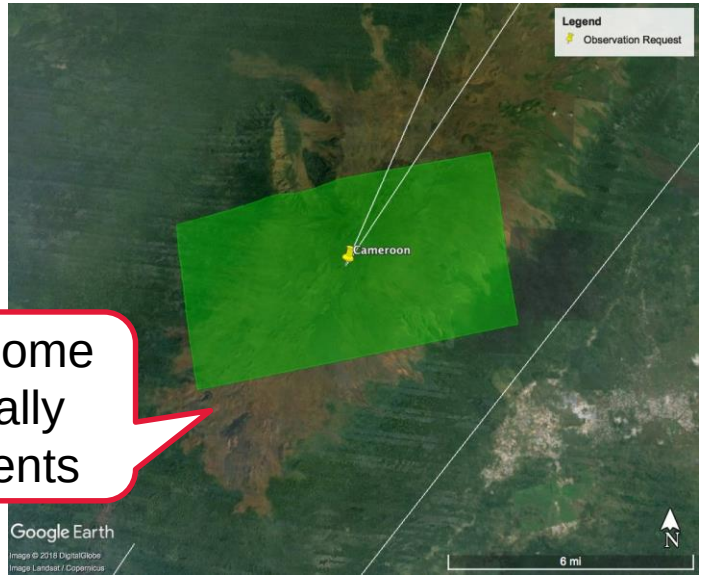

```
$ python ../eagle_eye/analysis/scripts/plotMemoryProfile.py -c 12800 sc_1_memory.csv
```

Answering Mission Design Questions

Do we actually do what we want (qualitatively)? vector_report.kml

Sufficient

Insufficient



Missed some historically active vents

Need a CONOPS change: draw a polygon that covers the whole area, schedule as an area mosaic.

Making videos

- Systems Toolkit (STK)
  - Export a .a file after scheduling using the outputSTKAttitudeProfile scheduling function
- Satellite Orbit Analysis Platform
  - Run the outputSpacecraftCKernels() scheduling function after scheduling
  - Hand-configure a new SOAP scenario
- Cosmographia
  - Run the outputCosmographia() scheduling function after scheduling

This tutorial

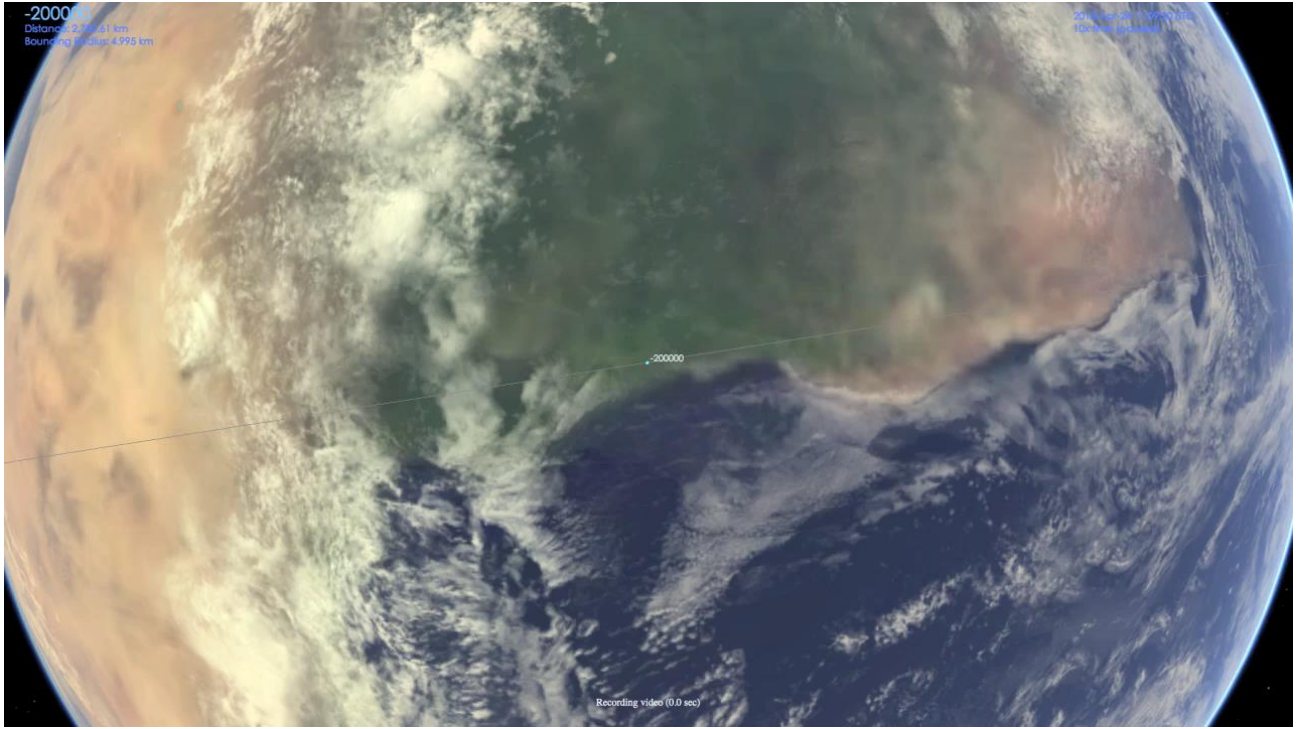
Launching Cosmographia

- Use the vector report (vector_report.csv)/Google Earth to identify a time of interest
- Launch Cosmographia

```
cd runCase_hypothetical_cosmo
./run_cosmographia.sh
```

- Set time to the start of the interesting time
- Customize view
  - Center on Earth
  - Rotate/turn
  - Turn on observer-frame path
  - Track the spacecraft (-200000)
- Set playback speed to 10x
- “File,” “Record.”
- Play immediately – there is no delay on recording
- Alternative: python scripting language interface

Example video



Observations from the Video

- This mission is actually quite easy: very few targets
  - Scientists will fill the gap with more requests, ask you to re-run the scheduler
- The spacecraft slews relatively often: may want to budget some design time for managing attitude actuators
- The spacecraft spends most of its time either idle or slewing
- Most of these observations happen off-nadir: images will be skewed and low quality
  - Feedback: increase target elevation angle constraints, re-run the scheduler

Jet Propulsion Laboratory
California Institute of Technology

jpl.nasa.gov