Closed Loop ComCam Image Ingestion and Application of Correction

This notebook is used to execute the [LVV-2229 (2.0)] test script during System Spread Integration Tests on Level 3.
It is part of the plan LVV-P81 and of the test cylce LVV-C176.
Execution steps are separated by horizontal lines.
Upon completion, save the notebook and its output as a pdf file to be attached to the test execution in JIRA.

LVV-T2229 (2.0) simply repeats the LVV-T2228 (1.0) test case twice, but with different targets.
This simulates two visits and tell us how the MTAOS behaves on sky.
The idea is that, depending on the angular distance between the two targets,
the MTAOS should use or not the corrections applied from the previous target.

In [1]:

```
from lsst.ts import utils

# Extract your name from the Jupyter Hub
__executed_by__ = os.environ["JUPYTERHUB_USER"]

# Extract execution date
__executed_on__ = utils.astropy_time_from_tai_unix(utils.current_tai())
__executed_on__.format = "isot"

# This is used later to define where Butler stores the images
summit = os.environ["LSST_DDS_PARTITION_PREFIX"] == "summit"

print(f"\nExecuted by {__executed_by__} on {__executed_on__}."
      f"\n  At the summit? {summit}")
```

```
lsst.ts.utils.tai INFO: Update leap second table
lsst.ts.utils.tai INFO: current_tai uses the system TAI clock
Executed by b1quint on 2022-04-08T16:34:40.336.
  At the summit? True
```

Initial Setup

log onto the summit nublado
https://summit-lsp.lsst.codes/
git clone the ts_notebook repo

There will be a series of procedures to set up, "slew" and track the telescope before we get an image.
This is similar to test case LVV-T2189.

Check ComCam Playback Mode

Verify that ComCam can be use the playback option and that the required images are stored in the right place **TBD**.

Load all the needed libraries

Using the setup procedure, get the remotes and the components ready.

This includes simulators as well as real hardware when available (this will depend on when the test is conducted at NCSA or on level 3 or on the telescope):

- pointing
- mount (with the CCW)
- rotator
- ready M1M3: raise mirror, turn on FB, clear forces. Note that if used at level 3, we need to have M1M3 LUT use mount telemetry
- ready M2: turn on FB, clear forces. Note that if used at level 3, we need to have M2 LUT use mount telemetry
- Get cam hex Ready: check config; make sure LUT is on and has valid inputs; make sure hex is at LUT position
- Get M2 hex (simulator) Ready: check config; make sure LUT is on and has valid inputs; make sure hex is at LUT position
- Finally, get the MTAOS CSC ready

In [2]:

```
%load_ext autoreload
%autoreload 2
```

In [3]:

```
import rubin_jupyter_utils.lab.notebook as nb
nb.utils.get_node()
```

```
/tmp/ipykernel_38936/1665379685.py:2: DeprecationWarning: Call to deprecated f
unction (or staticmethod) get_node. (Please use lsst.rsp.get_node())
  nb.utils.get_node()
```

Out[3]:

```
'yagan07'
```

In [4]:

```
import os
import sys
import asyncio
import logging

import pandas as pd
import numpy as np

from matplotlib import pyplot as plt
```

```
import lsst.daf.butler as dafButler

from lsst.ts import salobj
from lsst.ts.observatory.control.maintel import MTCS, ComCam
from lsst.ts.observatory.control import RotType
```

```
WARNING: version mismatch between CFITSIO header (v4.000999999999999) and link
ed library (v4.01).


WARNING: version mismatch between CFITSIO header (v4.000999999999999) and link
ed library (v4.01).


WARNING: version mismatch between CFITSIO header (v4.000999999999999) and link
ed library (v4.01).
```

In [5]:

```
logging.basicConfig(format="%(name)s:%(message)s", level=logging.DEBUG)
```

In [6]:

```
log = logging.getLogger("setup")
log.level = logging.DEBUG
```

In [7]:

```
domain = salobj.Domain()
```

In [8]:

```
mtcs = MTCS(domain=domain, log=log)
mtcs.set_rem_loglevel(40)
```

```
setup.MTCS DEBUG: mtmount: Adding all resources.
setup.MTCS DEBUG: mtptg: Adding all resources.
setup.MTCS DEBUG: mtaos: Adding all resources.
setup.MTCS DEBUG: mtm1m3: Adding all resources.
setup.MTCS DEBUG: mtm2: Adding all resources.
setup.MTCS DEBUG: mthexapod_1: Adding all resources.
setup.MTCS DEBUG: mthexapod_2: Adding all resources.
setup.MTCS DEBUG: mtrotator: Adding all resources.
setup.MTCS DEBUG: mtdome: Adding all resources.
setup.MTCS DEBUG: mtdometrajectory: Adding all resources.
```

In [9]:

```
await mtcs.start_task
```

```
MTHexapod INFO: Read historical data in 0.05 sec
MTHexapod INFO: Read historical data in 0.07 sec
MTHexapod.electrical WARNING: electrical DDS read queue is filling: 12 of 1
00 elements
MTHexapod.application WARNING: application DDS read queue is filling: 12 of
100 elements
```

Out[9]:

```
[None, None, None, None, None, None, None, None, None, None]
```

```
MTHexapod.actuators WARNING: actuators DDS read queue is filling: 13 of 100
elements
```

In [10]:

```
comcam = ComCam(domain=domain, log=log)
comcam.set_rem_loglevel(40)
```

```
setup.ComCam DEBUG: cccamera: Adding all resources.
setup.ComCam DEBUG: ccheaderservice: Adding all resources.
setup.ComCam DEBUG: ccarchiver: Adding all resources.
```

In [11]:
```
await comcam.start_task
```

Out[11]:
```
[None, None, None]
```

In [12]:
```
await comcam.enable()
```

```
setup.ComCam INFO: Enabling all components
setup.ComCam DEBUG: Gathering settings.
setup.ComCam DEBUG: Couldn't get settingVersions event. Using empty setting
s.
setup.ComCam DEBUG: Couldn't get settingVersions event. Using empty setting
s.
setup.ComCam DEBUG: Complete settings for cccamera.
setup.ComCam DEBUG: Complete settings for ccheaderservice.
setup.ComCam DEBUG: Complete settings for ccarchiver.
setup.ComCam DEBUG: Settings versions: {'cccamera': '', 'ccheaderservice':
 '', 'ccarchiver': ''}
setup.ComCam DEBUG: [cccamera]::[<State.ENABLED: 2>]
setup.ComCam DEBUG: [ccheaderservice]::[<State.ENABLED: 2>]
setup.ComCam DEBUG: [ccarchiver]::[<State.ENABLED: 2>]
setup.ComCam INFO: All components in <State.ENABLED: 2>.
```

In [17]:
```
script = salobj.Controller("Script", index=42658886)
await asyncio.sleep(10) # wait 10 second may help with DDS problems; closing al
```

```
Script INFO: Read historical data in 0.00 sec
```

In [18]:
```
from datetime import datetime
script.log.info(f"LVV-T2229 - START -- {datetime.now()} UTC")
```

```
Script INFO: LVV-T2229 - START -- 2022-04-08 16:40:07.132375 UTC
```

Slew and Track

Using the slew procedure, slew the systems to a specific elevation, azimuth and rotator angle. Verify that the telemetry is generated.

Slew to **RA 20:28:18.74** and **DEC -87:28:19.9** with **rot_type=RotType.Physical** and **Rotator Angle of 0°**. We use this field because it is the field that was simulated and that is a field that is visible the whole year.

RotType Physical Ensures that the Rotator will not move. This is necessary because the CCW is not running (MTmount in simulation mode).

Slew to target:

In [14]:
```
await mtcs.slew_icrs(ra="20:28:18.74", dec="-87:28:19.9", rot_type=RotType.Sky,
setup.MTCS DEBUG: RotSky = 180.0 deg, RotPhys = 69.28768768828525 deg.
setup.MTCS DEBUG: Wait 5.0s for rotator to settle down.
setup.MTCS DEBUG: Workaround for rotator trajectory problem. Moving rotator
to its current position: 1.64
setup.MTCS DEBUG: Wait for MTRotator in position event.
setup.MTCS DEBUG: MTRotator in position: True.
setup.MTCS DEBUG: MTRotator already in position. Handling potential race co
ndition.
setup.MTCS INFO: MTRotator in position: False.
setup.MTCS INFO: MTRotator in position: True.
setup.MTCS DEBUG: MTRotator in position True. Waiting settle time 5.0s
setup.MTCS DEBUG: Sending slew command.
setup.MTCS DEBUG: Scheduling check coroutines
setup.MTCS DEBUG: process as completed...
setup.MTCS DEBUG: Monitor position started.
setup.MTCS DEBUG: Waiting for Target event from mtmount.
setup.MTCS DEBUG: mtmount: <State.ENABLED: 2>
setup.MTCS DEBUG: mtptg: <State.ENABLED: 2>
setup.MTCS DEBUG: mtaos: <State.ENABLED: 2>
setup.MTCS DEBUG: mtm1m3: <State.ENABLED: 2>
setup.MTCS DEBUG: mtm2: <State.ENABLED: 2>
setup.MTCS DEBUG: mthexapod_1: <State.ENABLED: 2>
setup.MTCS DEBUG: mthexapod_2: <State.ENABLED: 2>
setup.MTCS DEBUG: mtrotator: <State.ENABLED: 2>
setup.MTCS DEBUG: mtdome: <State.ENABLED: 2>
setup.MTCS DEBUG: mtdometrajectory: <State.ENABLED: 2>
setup.MTCS DEBUG: Wait for mtmount in position events.
setup.MTCS DEBUG: Wait for dome in position event.
setup.MTCS DEBUG: Wait for MTRotator in position event.
setup.MTCS DEBUG: MTRotator in position: True.
setup.MTCS DEBUG: MTRotator already in position. Handling potential race co
ndition.
setup.MTCS DEBUG: Wait for MTMount elevation in position event.
setup.MTCS DEBUG: MTMount elevation in position: True.
setup.MTCS DEBUG: MTMount elevation already in position. Handling potential
race condition.
setup.MTCS DEBUG: Wait for MTMount azimuth in position event.
setup.MTCS DEBUG: MTMount azimuth in position: True.
setup.MTCS DEBUG: MTMount azimuth already in position. Handling potential r
ace condition.
setup.MTCS DEBUG: Mount target: private_revCode: bdcb00ba, private_sndStam
p: 1649435764.8681202, private_rcvStamp: 1649435764.8761108, private_seqNu
m: 24958, private_identity: MTMount, private_origin: 263534, elevation: 31.
3145690557216, elevationVelocity: -0.00017477235087341318, azimuth: 182.777
528446321, azimuthVelocity: 8.701946103498314e-05, taiTime: 1649435764.9259
527, trackId: 2, tracksys: SIDEREAL, radesys: ICRS, priority: 0
setup.MTCS INFO: MTMount elevation in position: False.
setup.MTCS INFO: MTMount azimuth in position: False.
setup.MTCS INFO: MTRotator in position: False.
setup.MTCS DEBUG: [Tel]: Az = +119.206[ +63.6]; El = +060.384[ -29.1] [Ro
t]: +001.638[  -0.0] [Dome] Az = +000.000; El = +000.000
setup.MTCS DEBUG: Dome azimuth in position.
setup.MTCS DEBUG: Dome elevation in position.
setup.MTCS DEBUG: [Tel]: Az = +155.291[ +27.5]; El = +042.338[ -11.0] [Ro
t]: -005.906[  +0.0] [Dome] Az = +000.000; El = +000.000
setup.MTCS WARNING: mtrotator not in <State.ENABLED: 2>: <State.FAULT: 3>
```

```
---------------------------------------------------------------------------
RuntimeError                              Traceback (most recent call last)
Input In [14], in <cell line: 1>()
----> 1 await mtcs.slew_icrs(ra="20:28:18.74", dec="-87:28:19.9", rot_type=Rot
Type.Sky, rot=180)

File ~/auto-op-env-packages/ts_observatory_control/python/lsst/ts/observatory/
control/base_tcs.py:590, in BaseTCS.slew_icrs(self, ra, dec, rot, rot_type, ta
rget_name, dra, ddec, offset_x, offset_y, az_wrap_strategy, time_on_target, sl
ew_timeout, stop_before_slew, wait_settle)
    585     valid_rottypes = ", ".join(repr(rt) for rt in RotType)
    586     raise RuntimeError(
    587         f"Unrecognized rottype {rot_type}. Should be one of {valid_rot
types}"
    588     )
--> 590 await self.slew(
    591     radec_icrs.ra.hour,
    592     radec_icrs.dec.deg,
    593     rotPA=rot_angle.deg,
    594     target_name=target_name,
    595     frame=self.CoordFrame.ICRS,
    596     epoch=2000,
    597     equinox=2000,
    598     parallax=0,
    599     pmRA=0,
    600     pmDec=0,
    601     rv=0,
    602     dRA=dra,
    603     dDec=ddec,
    604     rot_frame=rot_frame,
    605     rot_track_frame=rot_track_frame,
    606     az_wrap_strategy=az_wrap_strategy,
    607     time_on_target=time_on_target,
    608     rot_mode=self.RotMode.FIELD,
    609     slew_timeout=slew_timeout,
    610     stop_before_slew=stop_before_slew,
    611     wait_settle=wait_settle,
    612     offset_x=offset_x,
    613     offset_y=offset_y,
    614 )
    616 return radec_icrs, rot_angle

File ~/auto-op-env-packages/ts_observatory_control/python/lsst/ts/observatory/
control/base_tcs.py:762, in BaseTCS.slew(self, ra, dec, rotPA, target_name, fr
ame, epoch, equinox, parallax, pmRA, pmDec, rv, dRA, dDec, rot_frame, rot_trac
k_frame, rot_mode, az_wrap_strategy, time_on_target, slew_timeout, stop_before
_slew, wait_settle, offset_x, offset_y)
    755 getattr(self.rem, self.ptg_name).cmd_poriginOffset.set(
    756     dx=offset_x * self.plate_scale,
    757     dy=offset_y * self.plate_scale,
    758     num=0,
    759 )
    761 try:
--> 762     await self._slew_to(
    763         getattr(self.rem, self.ptg_name).cmd_raDecTarget,
    764         slew_timeout=slew_timeout,
    765         offset_cmd=getattr(self.rem, self.ptg_name).cmd_poriginOffset,
    766         stop_before_slew=stop_before_slew,
    767         wait_settle=wait_settle,
    768     )
    769 except salobj.AckError as ack_err:
    770     self.log.error(
    771         f"Command to track target {target_name} rejected: {ack_err.ack
cmd.result}"
    772     )

File ~/auto-op-env-packages/ts_observatory_control/python/lsst/ts/observatory/
control/maintel/mtcs.py:297, in MTCS._slew_to(self, slew_cmd, slew_timeout, of
fset_cmd, stop_before_slew, wait_settle, check)
    292         getattr(self.rem, comp).evt_summaryState.flush()
    293         self.scheduled_coro.append(
    294             asyncio.create_task(self.check_component_state(comp))
    295         )
--> 297 await self.process_as_completed(self.scheduled_coro)

File ~/auto-op-env-packages/ts_observatory_control/python/lsst/ts/observatory/
control/remote_group.py:1157, in RemoteGroup.process_as_completed(self, tasks)
   1155 except Exception as e:
   1156     await self.cancel_not_done(tasks)
-> 1157     raise e
   1158 else:
   1159     await self.cancel_not_done(tasks)

File ~/auto-op-env-packages/ts_observatory_control/python/lsst/ts/observatory/
control/remote_group.py:1154, in RemoteGroup.process_as_completed(self, tasks)
   1152 for res in asyncio.as_completed(tasks):
   1153     try:
-> 1154         ret_val = await res
   1155     except Exception as e:
   1156         await self.cancel_not_done(tasks)

File /opt/lsst/software/stack/conda/miniconda3-py38_4.9.2/envs/lsst-scipipe-3.
0.0/lib/python3.8/asyncio/tasks.py:619, in as_completed.<locals>._wait_for_one
()
    616 if f is None:
    617     # Dummy value from _on_timeout().
    618     raise exceptions.TimeoutError
--> 619 return f.result()

File ~/auto-op-env-packages/ts_observatory_control/python/lsst/ts/observatory/
control/remote_group.py:495, in RemoteGroup.check_component_state(self, compon
ent, desired_state)
    493 if state != desired_state:
    494     self.log.warning(f"{component} not in {desired_state!r}: {state!r}
")
--> 495     raise RuntimeError(
    496         f"{component} state is {state!r}, expected {desired_state!r}"
    497     )
    498 else:
    499     self.log.debug(f"{component}: {state!r}")

RuntimeError: mtrotator state is <State.FAULT: 3>, expected <State.ENABLED: 2>
```

In [15]:

```
await mtcs.slew_icrs(ra="20:28:18.74", dec="-87:28:19.9", rot_type=RotType.Sky,
```

```
setup.MTCS DEBUG: RotSky = 180.0 deg, RotPhys = 70.08438221591558 deg.
setup.MTCS DEBUG: Wait 5.0s for rotator to settle down.
setup.MTCS DEBUG: Workaround for rotator trajectory problem. Moving rotator
to its current position: -11.23
```

```
setup.MTCS DEBUG: Wait for MTRotator in position event.
setup.MTCS DEBUG: MTRotator in position: False.
setup.MTCS INFO: MTRotator in position: True.
setup.MTCS DEBUG: MTRotator in position True. Waiting settle time 5.0s
setup.MTCS DEBUG: Sending slew command.
setup.MTCS DEBUG: Scheduling check coroutines
setup.MTCS DEBUG: process as completed...
setup.MTCS DEBUG: Monitor position started.
setup.MTCS DEBUG: Waiting for Target event from mtmount.
setup.MTCS DEBUG: mtmount: <State.ENABLED: 2>
setup.MTCS DEBUG: mtptg: <State.ENABLED: 2>
setup.MTCS DEBUG: mtaos: <State.ENABLED: 2>
setup.MTCS DEBUG: mtm1m3: <State.ENABLED: 2>
setup.MTCS DEBUG: mtm2: <State.ENABLED: 2>
setup.MTCS DEBUG: mthexapod_1: <State.ENABLED: 2>
setup.MTCS DEBUG: mthexapod_2: <State.ENABLED: 2>
setup.MTCS DEBUG: mtrotator: <State.ENABLED: 2>
setup.MTCS DEBUG: mtdome: <State.ENABLED: 2>
setup.MTCS DEBUG: mtdometrajectory: <State.ENABLED: 2>
setup.MTCS DEBUG: Wait for mtmount in position events.
setup.MTCS DEBUG: Wait for dome in position event.
setup.MTCS DEBUG: Wait for MTRotator in position event.
setup.MTCS DEBUG: MTRotator in position: True.
setup.MTCS DEBUG: MTRotator already in position. Handling potential race co
ndition.
setup.MTCS DEBUG: Wait for MTMount elevation in position event.
setup.MTCS DEBUG: MTMount elevation in position: True.
setup.MTCS DEBUG: MTMount elevation already in position. Handling potential
race condition.
setup.MTCS DEBUG: Wait for MTMount azimuth in position event.
setup.MTCS DEBUG: MTMount azimuth in position: True.
setup.MTCS DEBUG: MTMount azimuth already in position. Handling potential r
ace condition.
setup.MTCS DEBUG: Mount target: private_revCode: bdcb00ba, private_sndStam
p: 1649435953.1138284, private_rcvStamp: 1649435953.114719, private_seqNum:
25145, private_identity: MTMount, private_origin: 263534, elevation: 31.281
573585471435, elevationVelocity: -0.00017578499726828795, azimuth: 182.7936
3911739378, azimuthVelocity: 8.414500971222532e-05, taiTime: 1649435953.170
3763, trackId: 3, tracksys: SIDEREAL, radesys: ICRS, priority: 0
setup.MTCS INFO: MTMount elevation in position: False.
setup.MTCS INFO: MTMount azimuth in position: False.
setup.MTCS INFO: MTRotator in position: False.
setup.MTCS INFO: MTMount elevation in position: True.
setup.MTCS DEBUG: MTMount elevation in position True. Waiting settle time
 3.0s
setup.MTCS INFO: MTMount azimuth in position: True.
setup.MTCS DEBUG: MTMount azimuth in position True. Waiting settle time 3.0
s
setup.MTCS DEBUG: [Tel]: Az = +181.662[  +1.1]; El = +031.381[  -0.1] [Ro
t]: -011.233[  -0.0] [Dome] Az = +000.000; El = +000.000
```

```
setup.MTCS DEBUG: Dome azimuth in position.
setup.MTCS DEBUG: Dome elevation in position.
setup.MTCS DEBUG: [Tel]: Az = +182.794[  +0.0]; El = +031.280[  -0.0] [Ro
t]: -026.083[  -0.0] [Dome] Az = +000.000; El = +000.000
setup.MTCS DEBUG: [Tel]: Az = +182.795[  +0.0]; El = +031.279[  -0.0] [Ro
t]: -047.107[  -0.0] [Dome] Az = +000.000; El = +000.000
setup.MTCS DEBUG: [Tel]: Az = +182.795[  +0.0]; El = +031.278[  -0.0] [Ro
t]: -067.185[  -0.0] [Dome] Az = +000.000; El = +000.000
setup.MTCS INFO: MTRotator in position: True.
setup.MTCS DEBUG: MTRotator in position True. Waiting settle time 3.0s
setup.MTCS DEBUG: [Tel]: Az = +182.796[  +0.0]; El = +031.277[  -0.0] [Ro
t]: -069.992[  -0.0] [Dome] Az = +000.000; El = +000.000
```

```
Out[15]: (<ICRS Coordinate: (ra, dec) in deg
              (307.07808333, -87.47219444)>,
          <Angle 180. deg>)
```

Take in-focus image

Once the different components are ready (M1M3, M2, rotator and CCW, hexapods) and tracking, take an image using the take_image command in playback mode.
This second image should be the one that uses the correction calculated with the first slew.

```
In [16]: exp_focus = await comcam.take_object(15)
         print(f"Target exposure: {exp_focus}")
```

```
setup.ComCam DEBUG: Generating group_id
setup.ComCam DEBUG: imagetype: OBJECT, TCS synchronization not configured.
setup.ComCam DEBUG: OBJECT 0001 - 0001
Target exposure: [2022040800007]
```

Intra Focus Position

Using the Camera Hexapod, piston ComCam +1mm

```
In [19]: await mtcs.rem.mthexapod_1.cmd_offset.set_start(z=1000.)
```

```
Out[19]: <ddsutil.MTHexapod_ackcmd_c4d6958b at 0x7fef739c46a0>
```

Intra Focus Image

While tracking, take an image with ComCam and check that the header is containing the right telemetry

```
In [20]: exp_intra = await comcam.take_object(15)
```

```
print(f"Target 1 exposure: {exp_intra}")
```

```
setup.ComCam DEBUG: Generating group_id
setup.ComCam DEBUG: imagetype: OBJECT, TCS synchronization not configured.
setup.ComCam DEBUG: OBJECT 0001 - 0001
Target 1 exposure: [2022040800008]
```

Extra Focus Position

Using the Camera Hexapod, piston ComCam to -1mm

In [21]:

```
await mtcs.rem.mthexapod_1.cmd_offset.set_start(z=-2000.)
```

Out[21]:

```
<ddsutil.MTHexapod_ackcmd_c4d6958b at 0x7fefa6b8fd60>
```

Extra Focus Image

While tracking, take an image with ComCam and check that the header is containing the right telemetry.

In [22]:

```
exp_extra = await comcam.take_object(15)
print(f"Target 1 exposure: {exp_extra}")
```

```
setup.ComCam DEBUG: Generating group_id
setup.ComCam DEBUG: imagetype: OBJECT, TCS synchronization not configured.
setup.ComCam DEBUG: OBJECT 0001 - 0001
Target 1 exposure: [2022040800009]
```

Go Back to Focus Position

Put the hexapod back to 0mm.

In [23]:

```
await mtcs.rem.mthexapod_1.cmd_offset.set_start(z=1000.)
```

Out[23]:

```
<ddsutil.MTHexapod_ackcmd_c4d6958b at 0x7feff6212670>
```

Stop Tracking

If using MTMount Simulator and CCW Following Mode Disabled, stop tracking to prevent the Rotator to hit the limit switches.

In [24]:

```
await mtcs.stop_tracking()
```

```
setup.MTCS DEBUG: Stop tracking.
```

Get Zernike Coefficients

Use the MTAOS Wavefront Estimator Pipeline to calculate the required Zernike Coefficients that represent the Wavefront data.

In [25]:

```
import yaml

wep_config = yaml.safe_dump(
    dict(
        tasks=dict(
            isr=dict(
                config=dict(
                    doOverscan=False,
                    doApplyGains=False,
                )
            ),
            generateDonutCatalogWcsTask=dict(
        config={
            "filterName": "phot_g_mean",
            "connections.refCatalogs": "gaia_dr2_20200414",
            "donutSelector.sourceLimit": 10,
            "donutSelector.fluxField": "phot_g_mean_flux"
            }
        )
    )
)
)
```

In [26]:

```
await mtcs.rem.mtaos.cmd_runWEP.set_start(visitId=exp_intra[0] - 2021111900000,
                                          extraId=exp_extra[0] - 2021111900000)
```

```
---------------------------------------------------------------------------
AckError                                  Traceback (most recent call last)
Input In [26], in <cell line: 1>()
----> 1 await mtcs.rem.mtaos.cmd_runWEP.set_start(visitId=exp_intra[0] - 20211
11900000,
      2                                         extraId=exp_extra[0] - 20211
11900000)

File /opt/lsst/software/stack/conda/miniconda3-py38_4.9.2/envs/lsst-scipipe-3.
0.0/lib/python3.8/site-packages/lsst/ts/salobj/topics/remote_command.py:423, i
n RemoteCommand.set_start(self, timeout, wait_done, **kwargs)
    382 """Create a new ``self.data``, set zero or more fields,
    383 and start the command.
    384 
   (...)
    420     If ``data`` is not None and not an instance of `DataType`.
    421 """
    422 self.set(**kwargs)
--> 423 return await self.start(timeout=timeout, wait_done=wait_done)

File /opt/lsst/software/stack/conda/miniconda3-py38_4.9.2/envs/lsst-scipipe-3.
0.0/lib/python3.8/site-packages/lsst/ts/salobj/topics/remote_command.py:483, i
n RemoteCommand.start(self, data, timeout, wait_done)
    479 cmd_info = CommandInfo(
    480     remote_command=self, seq_num=seq_num, wait_done=wait_done
    481 )
    482 self.salinfo._running_cmds[seq_num] = cmd_info
--> 483 return await cmd_info.next_ackcmd(timeout=timeout)

File /opt/lsst/software/stack/conda/miniconda3-py38_4.9.2/envs/lsst-scipipe-3.
0.0/lib/python3.8/site-packages/lsst/ts/salobj/topics/remote_command.py:201, i
n CommandInfo.next_ackcmd(self, timeout)
    199     ackcmd = await self._wait_task
    200     if ackcmd.ack in self.failed_ack_codes:
--> 201         raise base.AckError(msg="Command failed", ackcmd=ackcmd)
    202     return ackcmd
    203 except asyncio.TimeoutError:

AckError: msg='Command failed', ackcmd=(ackcmd private_seqNum=1414931397, ack=
<SalRetCode.CMD_FAILED: -302>, error=1, result='Failed: Error running pipeline
task: ')
```

Get Corrections

Use the MTAOS Optical Feedback Controller to retrieve the corrections that should be applied to m1m3, m2, camera hexapod, and m2 hexapod.

In [ ]:

```
await mtcs.rem.mtaos.cmd_runOFC.start(timeout=60.)
```

Issue the corrections

Issue the corrections found by the MTAOS OFC to m1m3, m2, camera hexapod, and m2 hexapod.

In [ ]:

```
await mtcs.rem.mtaos.cmd_issueCorrection.start(timeout=60.)
```

Verify ISR Data

Make sure that the Instrument Signature Removal ran on the intra- and extra-focus data and that this data is accessible via Butler.

In [ ]:

```
if summit:
    butler = dafButler.Butler("/repo/LSSTComCam/")
else:
    butler = dafButler.Butler("/repo/main/")
```

In [ ]:

```
registry = butler.registry

collections = [collection for collection in registry.queryCollections()
               if collection.startswith('mtaos_wep')]
```

In [ ]:

```
exp_intra_id = {'instrument': 'LSSTComCam',
                'detector': 0,
                'exposure': exp_intra[0]}

raw_intra = butler.get('postISRCCD', dataId=exp_intra_id,
                       collections=collections)

print(raw_intra.getMetadata())
```

In [ ]:

```
%matplotlib inline
fig, ax = plt.subplots(num="Intra Focus Image", figsize=(7,7), dpi=90)

vmin = np.percentile(raw_intra.image.array, 2)
vmax = np.percentile(raw_intra.image.array, 98)

ax.imshow(raw_intra.image.array,
          origin='lower',
          interpolation='nearest',
          vmin=vmin,
          vmax=vmax)
ax.set_xlabel("X [px]")
ax.set_ylabel("Y [px]")

fig.suptitle(f"Intra Focus Image\n{exp_intra_id['exposure']}")
fig.tight_layout()

plt.show()
```

In [ ]:

```
exp_extra_id = {'instrument': 'LSSTComCam',
                'detector': 0,
                'exposure': exp_extra[0]}

exp_extra = butler.get('postISRCCD', dataId=exp_extra_id,
                           collections=collections)

print(exp_extra.getMetadata())
```

In [ ]:

```
%matplotlib inline
fig, ax = plt.subplots(num="Extra Focus Image", figsize=(7, 7), dpi=90)

vmin = np.percentile(exp_extra.image.array, 2)
vmax = np.percentile(exp_extra.image.array, 98)

ax.imshow(exp_extra.image.array,
          origin='lower',
          interpolation='nearest',
          vmin=vmin,
          vmax=vmax)

ax.set_xlabel("X [px]")
ax.set_ylabel("Y [px]")

fig.suptitle(f"Extra Focus Image\n{exp_extra_id['exposure']}")
fig.tight_layout()

plt.show()
```

Slew and Track Second Target

Now, slew to a second target. The coordinates for this targets are **TBD** and depend on new simulated data. You will probably not run this for now until we have new simulated data. We will leave the notebook simply to have the structure pre-define.

Slew to **RA TBD** and **DEC TBD** with **rot_type=RotType.Physical** and **Rotator Angle of 0°**.

RotType Physical Ensures that the Rotator will not move. This is necessary because the CCW is not running (MTmount in simulation mode).

In [ ]:

```
await mtcs.slew_icrs(ra=???, dec=???, rot_type=RotType.Physical, rot=0)
```

Take in-focus image 2

Once the different components are ready (M1M3, M2, rotator and CCW, hexapods) and tracking, take an image using the take_image command in playback mode. This second image should be the one that uses the correction calculated with the first slew.

In [ ]:

```
exp_focus2 = await comcam.take_object(15)
print(f"Target exposure: {exp_focus2}")
```

Intra Focus Position 2

Using the Camera Hexapod, piston ComCam +1mm.

In [ ]:
```
await mtcs.rem.mthexapod_1.cmd_offset.set_start(z=1000.)
```

Intre Focus Image 2

While tracking, take an image and check that the header is containing the right telemetry.

In [ ]:
```
exp_intra2 = await comcam.take_object(15)
print(f"Target 1 exposure: {exp_intra2}")
```

Extra Focus Position 2

Apply an offset of -2000 um to the Camera Hexapod, to bring it down to -1 mm.

In [ ]:
```
await mtcs.rem.mthexapod_1.cmd_offset.set_start(z=-2000.)
```

Extra Focus Image 2

While tracking, take an image and check that the header is containing the right telemetry

In [ ]:
```
exp_extra2 = await comcam.take_object(15)
print(f"Target 1 exposure: {exp_extra2}")
```

Go back to focus position 2

Send the hexapod back to 0 mm by applying an offset of 1000 um in Z.

In [ ]:
```
await mtcs.rem.mthexapod_1.cmd_offset.set_start(z=1000.)
```

Stop tracking 2

If using MTMount Simulator and CCW Following Mode Disabled, stop tracking to prevent the Rotator to hit the limit switches.

```
In [ ]: await mtcs.stop_tracking()
```

Get Zernikes Coefficients 2

Use the MTAOS to calculate the required offsets to be sent to M1M3, M2, and the hexapods.

When we run the command in the example code below, if it does not raise the **TBD** error, then we know that the MTAOS WEP could find and retrieve the calibration files.

```
In [ ]: await mtcs.rem.mtaos.cmd_runWEP.set_start(visitId=exp_intra2[0],
                                                  extraId=exp_extra2[0])
```

Get Corrections 2

Apply the resulting offsets to the M1M3, M2 and the hexapods

```
In [ ]: await mtcs.rem.mtaos.cmd_runOFC.start(timeout=60.)
```

Issue the corrections 2

Issue (apply) the corrections found by the MTAOS OFC to m1m3, m2, camera hexapod, and m2 hexapod.

```
In [ ]: await mtcs.rem.mtaos.cmd_issueCorrection.start(timeout=60.)
```

Verify Offsets TBD</h2>

Verify that the offsets are the expected one by plotting:

- m1m3 actuator 101 z force
- m2 actuator B1 force
- camera hex y position
- m2 hex y position
- What about others?

Wrap Up and Shut Down

This section is intended for shutting down the system and should not be run as part of the regular testing procedure. Only run the following cells if you are done with the system and don't plan on executing any further tests.

```
In [ ]: await mtcs.set_state(salobj.State.STANDBY, components=["mtaos"])
```

```
In [ ]: await mtcs.lower_m1m3()
```

```
In [ ]: await mtcs.set_state(salobj.State.STANDBY, components=["mtm1m3"])
```

```
In [ ]: await mtcs.set_state(salobj.State.STANDBY, components=["mtm2"])
```

```
In [ ]: await mtcs.set_state(salobj.State.STANDBY, components=["mthexapod_1"])
```

```
In [ ]: await mtcs.set_state(salobj.State.STANDBY, components=["mthexapod_2"])
```

```
In [ ]: await mtcs.standby()
```

```
In [ ]: await comcam.standby()
```