LVV-T2229-Copy1

January 25, 2022

1 Closed Loop ComCam Image Ingestion and Application of Correction

This notebook is used for the level 3 integration tests from test plan LVV-P81 (https://jira.lsstcorp.org/secure/Tests.jspa#/testPlan/LVV-P81) as part of test cylce LVV-C176 (https://jira.lsstcorp.org/secure/Tests.jspa#/testCycle/LVV-C176). The following tests are currently run as part of this notebook:

- LVV-T2229 (https://jira.lsstcorp.org/secure/Tests.jspa#/testCase/LVV-T2229)

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.

Last executed by B. Quint

Run the setup.ipnyb notebook to bring all components up and in their enabled position. Check Chronograph.

Bring ComCom online and tranistion it to EnabledState. Check Chronograph.

[1]:
```
%load_ext autoreload
%autoreload 2
```

[2]:
```
import rubin_jupyter_utils.lab.notebook as nb
nb.utils.get_node()
```

```
/tmp/ipykernel_14870/1665379685.py:2: DeprecationWarning: Call to deprecated
function (or staticmethod) get_node. (Please use lsst.rsp.get_node())
  nb.utils.get_node()
```

[2]:
```
'yagan02'
```

[3]:
```
import os
import sys
import asyncio
import logging

import pandas as pd
```

```
import numpy as np

from matplotlib import pyplot as plt

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

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

[4]:
```
logging.basicConfig(format="%(name)s:%(message)s", level=logging.DEBUG)
```

[5]:
```
log = logging.getLogger("setup")
log.level = logging.DEBUG
```

[6]:
```
domain = salobj.Domain()
```

[7]:
```
mtcs = MTCS(domain=domain, log=log)
mtcs.set_rem_loglevel(40)
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

[8]:
```
await mtcs.start_task
```

[8]: [None, None, None, None, None, None, None, None, None, None]

[9]:
```
comcam = ComCam(domain=domain, log=log)
```

```
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
```

[10]:
```
comcam.set_rem_loglevel(40)
```

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

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

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

```
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
```

Find a target around az = 120º and el = 60º and rotator angle at PhysicalSky and 1.8º.

At this position, the rotator stays within a couple of degrees of its initial position. This is because the CCW is not running (MTmount in simulation mode).

target -> az = 120^o, el = 60^o

[12]:
```
target = await mtcs.find_target(az=120, el=60, mag_limit=8)

print(f"Target: {target}")
```

Target: HD 197094

[14]:
```
await mtcs.disable_ccw_following()
```

```
<IPython.core.display.HTML object>
```

[17]:
```
await mtcs.slew_icrs(ra="20:28:18.74", dec="-87:28:19.9", rot_type=RotType.
↪PhysicalSky, rot=2)
```

```
<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
```

[17]:
```
(<ICRS Coordinate: (ra, dec) in deg
     (307.07808333, -87.47219444)>,
 <Angle 2. deg>)
```

[18]:
```
await mtcs.stop_tracking()
```

```
<IPython.core.display.HTML object>
```

Slew to target:

[19]:
```
await mtcs.slew_icrs(ra="20:28:18.74", dec="-87:28:19.9", rot_type=RotType.
↪PhysicalSky, rot=2)
```

```
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>

<IPython.core.display.HTML object>
```

[19]:
```
(<ICRS Coordinate: (ra, dec) in deg
     (307.07808333, -87.47219444)>,
 <Angle 2. deg>)
```

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.

[20]:
```
exp_focus = await comcam.take_object(15)
print(f"Target exposure: {exp_focus}")
```

```
<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

Target exposure: [2022012500005]
```

Using the Camera Hexapod, piston ComCam +1mm

[21]:
```
await mtcs.move_camera_hexapod(x=0, y=0, z=+1000., u=0, v=0)
```

```
<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>
```

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

[22]:
```
exp_intra = await comcam.take_object(15)
print(f"Target 1 exposure: {exp_intra}")
```

```
<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

Target 1 exposure: [2022012500006]
```

Using the Camera Hexapod, piston ComCam to -1mm

[23]:
```
await mtcs.move_camera_hexapod(x=0, y=0, z=-1000., u=0, v=0)
```

```
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
```

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

[24]:
```
exp_extra = await comcam.take_object(15)
print(f"Target 1 exposure: {exp_extra}")
```

```
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
Target 1 exposure: [2022012500007]
```

Put the hexapod back to 0mm.

[25]:
```
await mtcs.move_camera_hexapod(x=0, y=0, z=0, u=0, v=0)
```

```
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
<IPython.core.display.HTML object>
```

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

[26]:
```
await mtcs.stop_tracking()
```

```
<IPython.core.display.HTML object>
```

[27]:
```
import yaml
```

[31]:
```
run_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"
                }
            )
        )
    )
)
```

[32]:
```
await mtcs.rem.mtaos.cmd_setLogLevel.set_start(level=10)
```

[32]:
```
<lsst.ts.salobj._ddsutil.MTAOS_ackcmd_fd03e870 at 0x7ff688929a30>
```

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

[33]:
```
await mtcs.rem.mtaos.cmd_runWEP.set_start(visitId=exp_intra[0] - 2021111900000,
                                          extraId=exp_extra[0] - 2021111900000,
                                          config=run_wep_config)
```

```
---------------------------------------------------------------------------
AckError                                  Traceback (most recent call last)
/tmp/ipykernel_14870/3273614236.py in <module>
----> 1 await mtcs.rem.mtaos.cmd_runWEP.set_start(visitId=exp_intra[0] -␣
 ↪2021111900000,
      2                                           extraId=exp_extra[0] -␣
 ↪2021111900000,
      3                                           config=run_wep_config)

/opt/lsst/software/stack/conda/miniconda3-py38_4.9.2/envs/lsst-scipipe-0.7.0/li /
 ↪python3.8/site-packages/lsst/ts/salobj/topics/remote_command.py in␣
 ↪set_start(self, timeout, wait_done, **kwargs)
    421         """
    422         self.set(**kwargs)
```

```
--> 423         return await self.start(timeout=timeout, wait_done=wait_done)
    424
    425     async def start(

/opt/lsst/software/stack/conda/miniconda3-py38_4.9.2/envs/lsst-scipipe-0.7.0/li /
 ↪python3.8/site-packages/lsst/ts/salobj/topics/remote_command.py in start(self ␣
 ↪data, timeout, wait_done)
    481         )
    482         self.salinfo._running_cmds[seq_num] = cmd_info
--> 483         return await cmd_info.next_ackcmd(timeout=timeout)

/opt/lsst/software/stack/conda/miniconda3-py38_4.9.2/envs/lsst-scipipe-0.7.0/li /
 ↪python3.8/site-packages/lsst/ts/salobj/topics/remote_command.py in␣
 ↪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=1414931398,␣
 ↪ack=<SalRetCode.CMD_FAILED: -302>, error=1, result="Failed: Error running␣
 ↪pipeline task:   def selectSources(self, sourceCat, bbox):\n\nlsst.pipe.base.
 ↪graphBuilder WARNING: No dimension records for element 'visit' found.\npy.
 ↪warnings WARNING: /opt/lsst/software/stack/stack/miniconda3-py38_4.9.2-0.7.0/
 ↪Linux64/c")
```

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

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

[ ]:
```
await mtcs.set_state(
    state=salobj.State.STANDBY,
    settings=dict(mtaos="impg"),
    components=["mtaos"]
    )
```

[ ]:
```
await mtcs.set_state(
    state=salobj.State.ENABLED,
    settings=dict(mtaos="impg"),
    components=["mtaos"]
    )
```

Process wavefront data

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

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

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

Issue the corrections

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

Query the butler to verify that the images are there and check the metadata. This step must be verified using a separate noteboook.

1.1 Wrap Up and Shut Down

This cell is not currently included as part of the test execution, but included here as needed to shutdown the systems

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

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

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

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

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

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

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

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

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

[ ]:
```

```