LVV-T2193

March 2, 2022

1 MTAOS handling of rejected commands

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-T2193 (https://jira.lsstcorp.org/secure/Tests.jspa#/testCase/LVV-T2193)

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 updated by E. Dennihy 20211020

Load all the needed libraries. Get the remotes ready Code in the notebook including section: "Check the summary state of each CSC".

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

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

```
/tmp/ipykernel_49586/1665379685.py:2: DeprecationWarning: Call to deprecated
function (or staticmethod) get_node. (Please use lsst.rsp.get_node())
  nb.utils.get_node()
```

[2]:
```
'yagan07'
```

[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>
```

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

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

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

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

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

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

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

Ready M1M3: Raise mirror, turn on FB, clear forces

Need to have M1M3 LUT use its inclinometer.

Ready M2: Turn on FB, clear forces

Need to have M2 LUT use its inclinometer

Get camera hexapod ready: check config; make sure LUT is on, and has valid inputs; make sure hex is at LUT position

---

Get M2 hexapod ready: check config; make sure LUT is on, and has valid inputs; make sure hex is at LUT position

---

Slew to the next target. Choose a target such that the rotator stays within a couple of degrees of its initial position. This is because the CCW is not running (MTmount in simulation mode).

[9]:
```
target = await mtcs.find_target(el=60, az=120, mag_limit=8)
print(target)
```

```
HD  12069
```

[10]:
```
await mtcs.slew_object(target, rot_type=RotType.PhysicalSky, rot=1.9)
```

```
<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>
```

```
---------------------------------------------------------------------------
RuntimeError                              Traceback (most recent call last)
Input In [10], in <module>
----> 1 await mtcs.slew_object(target, rot_type=RotType.PhysicalSky, rot=1.9)

File ~/auto-op-env-packages/ts_observatory_control/python/lsst/ts/observatory/
  ↪control/base_tcs.py:415, in BaseTCS.slew_object(self, name, rot, rot_type,␣
  ↪dra, ddec, offset_x, offset_y, az_wrap_strategy, time_on_target, slew_timeout
    411 object_table = self.object_list_get(name)
    413 self.log.info(f"Slewing to {name}: {object_table['RA']}␣
  ↪{object_table['DEC']}")
--> 415 await self.slew_icrs(
    416     ra=object_table["RA"],
    417     dec=object_table["DEC"],
    418     rot=rot,
    419     rot_type=rot_type,
```

```
    420     target_name=name,
    421     dra=dra,
    422     ddec=ddec,
    423     offset_x=offset_x,
    424     offset_y=offset_y,
    425     az_wrap_strategy=az_wrap_strategy,
    426     time_on_target=time_on_target,
    427     slew_timeout=slew_timeout,
    428 )

File ~/auto-op-env-packages/ts_observatory_control/python/lsst/ts/observatory/
 ↪control/base_tcs.py:589, in BaseTCS.slew_icrs(self, ra, dec, rot, rot_type,␣
 ↪target_name, dra, ddec, offset_x, offset_y, az_wrap_strategy, time_on_target,␣
 ↪slew_timeout, stop_before_slew, wait_settle)
    584     valid_rottypes = ", ".join(repr(rt) for rt in RotType)
    585     raise RuntimeError(
    586         f"Unrecognized rottype {rot_type}. Should be one of␣
 ↪{valid_rottypes}"
    587     )
--> 589 await self.slew(
    590     radec_icrs.ra.hour,
    591     radec_icrs.dec.deg,
    592     rotPA=rot_angle.deg,
    593     target_name=target_name,
    594     frame=self.CoordFrame.ICRS,
    595     epoch=2000,
    596     equinox=2000,
    597     parallax=0,
    598     pmRA=0,
    599     pmDec=0,
    600     rv=0,
    601     dRA=dra,
    602     dDec=ddec,
    603     rot_frame=rot_frame,
    604     rot_track_frame=rot_track_frame,
    605     az_wrap_strategy=az_wrap_strategy,
    606     time_on_target=time_on_target,
    607     rot_mode=self.RotMode.FIELD,
    608     slew_timeout=slew_timeout,
    609     stop_before_slew=stop_before_slew,
    610     wait_settle=wait_settle,
    611     offset_x=offset_x,
    612     offset_y=offset_y,
    613 )
    615 return radec_icrs, rot_angle
```

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

File ~/auto-op-env-packages/ts_observatory_control/python/lsst/ts/observatory/
 ↪control/maintel/mtcs.py:292, in MTCS._slew_to(self, slew_cmd, slew_timeout,␣
 ↪offset_cmd, stop_before_slew, wait_settle, check)
    287         getattr(self.rem, comp).evt_summaryState.flush()
    288         self.scheduled_coro.append(
    289             asyncio.create_task(self.check_component_state(comp))
    290         )
--> 292 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-2.0
 ↪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,␣
 ↪component, 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>
```

[11]:
```
await mtcs.set_state(salobj.State.ENABLED, components=["mtptg"])
```

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

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

[12]:
```
await mtcs.set_state(salobj.State.ENABLED, components=["mtrotator"])
```

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

```
---------------------------------------------------------------------------
RuntimeError                              Traceback (most recent call last)
Input In [12], in <module>
----> 1 await mtcs.set_state(salobj.State.ENABLED, components=["mtrotator"])

File ~/auto-op-env-packages/ts_observatory_control/python/lsst/ts/observatory/
 ↪control/remote_group.py:812, in RemoteGroup.set_state(self, state, settings,␣
 ↪components)
    809         self.log.debug(f"[{comp}]::{ret_val[i]!r}")
    811 if error_flag:
--> 812     raise RuntimeError(
    813         f"Failed to transition {failed_components} to "
    814         f"{salobj.State(state)!r}."
    815     )
    816 else:
    817     self.log.info(f"All components in {salobj.State(state)!r}.")
```

```
RuntimeError: Failed to transition ['mtrotator'] to <State.ENABLED: 2>.
```

Just run again...

[13]:
```
await mtcs.set_state(salobj.State.ENABLED, components=["mtrotator"])
```

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

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

[14]:
```
target = await mtcs.find_target(el=60, az=120, mag_limit=8)
print(target)
```

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

```
HD  12069
```

[15]:
```
await mtcs.slew_object(target, rot_type=RotType.PhysicalSky, rot=1.9)
```

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

---

clear all corrections using cmd_resetCorrection

[16]:
```
await mtcs.rem.mtaos.cmd_resetCorrection.start()
```

[16]:
```
<ddsutil.MTAOS_ackcmd_fd03e870 at 0x7f7fcef2f6a0>
```

[17]:
```
await mtcs.rem.mtaos.cmd_issueCorrection.start(timeout=60.)
```

[17]:
```
<ddsutil.MTAOS_ackcmd_fd03e870 at 0x7f7fcedb63a0>
```

Add 1um of z7 to the system via OFC, issue the corrections.

Compare the corrections sent vs forces and position changes applied. This is currently done in a separate notebook or on Chronograf.

[18]:
```
wavefront_errors = np.zeros(19)
```

[19]:
```
wavefront_errors[3]=1.0
```

[20]:
```
await mtcs.rem.mtaos.cmd_addAberration.set_start(wf=wavefront_errors,␣
↪timeout=10)
```

[20]:
```
<ddsutil.MTAOS_ackcmd_fd03e870 at 0x7f7fd7034490>
```

[21]:
```
await mtcs.rem.mtaos.cmd_issueCorrection.start(timeout=60.)
```

[21]:
```
<ddsutil.MTAOS_ackcmd_fd03e870 at 0x7f7fceb98e80>
```

Make plots using telemetry from each component to verify the changes in the DOFs. This step does not currently involve running any commands in this notebook. This step must be verified using a separate noteboook.

Put M2 hexapod in DISABLED state (so that we can test command rejection).

[22]:
```
await mtcs.set_state(salobj.State.DISABLED, components=["mthexapod_2"])
```

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

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

Add 1um of z7 to the system via OFC. Expect m2 hexapod corrections are rejected, and all other corrections applied, then undone.

[23]:
```
await mtcs.rem.mtaos.cmd_addAberration.set_start(wf=wavefront_errors,␣
↪timeout=10)
```

[23]:
```
<ddsutil.MTAOS_ackcmd_fd03e870 at 0x7f7fbdb74040>
```

[24]:
```
await mtcs.rem.mtaos.cmd_issueCorrection.start(timeout=60.)
```

```
---------------------------------------------------------------------------
AckError                                  Traceback (most recent call last)
Input In [24], in <module>
----> 1 await mtcs.rem.mtaos.cmd_issueCorrection.start(timeout=60.)
```

```
File /opt/lsst/software/stack/conda/miniconda3-py38_4.9.2/envs/lsst-scipipe-2.0
 ↪0/lib/python3.8/site-packages/lsst/ts/salobj/topics/remote_command.py:483, in
 ↪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-2.0
 ↪0/lib/python3.8/site-packages/lsst/ts/salobj/topics/remote_command.py:201, in
 ↪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=640532057,␣
 ↪ack=<SalRetCode.CMD_FAILED: -302>, error=1, result="Failed: Failed to apply␣
 ↪correction to: ['m2hex']. ")
```

Re-enable M2 hexapod Make it ready for AOS

[25]:
```
await mtcs.set_state(salobj.State.ENABLED, components=["mthexapod_2"])
```

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

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

Re-issue the correction.

[26]:
```
await mtcs.rem.mtaos.cmd_addAberration.set_start(wf=wavefront_errors,␣
 ↪timeout=10)
```

[26]:
```
<ddsutil.MTAOS_ackcmd_fd03e870 at 0x7f7fcefa5d30>
```

[27]:
```
await mtcs.rem.mtaos.cmd_issueCorrection.start(timeout=60.)
```

[27]:
```
<ddsutil.MTAOS_ackcmd_fd03e870 at 0x7f80346c7760>
```

Reject the latest corrections.

[28]:
```
await mtcs.rem.mtaos.cmd_rejectCorrection.start()
```

[28]:
```
<ddsutil.MTAOS_ackcmd_fd03e870 at 0x7f7fcef2ff10>
```

[29]:
```
await mtcs.rem.mtaos.cmd_issueCorrection.start(timeout=60.)
```

[29]:
```
<ddsutil.MTAOS_ackcmd_fd03e870 at 0x7f7fbda33250>
```

Add 2um of z7 via OFC

[30]:
```
wavefront_errors[3] = 2.0
```

[31]:
```
wavefront_errors
```

[31]:
```
array([0., 0., 0., 2., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
       0., 0.])
```

[32]:
```
await mtcs.rem.mtaos.cmd_addAberration.set_start(wf=wavefront_errors,␣
↪timeout=10)
```

[32]:
```
<ddsutil.MTAOS_ackcmd_fd03e870 at 0x7f7face11e20>
```

[33]:
```
await mtcs.rem.mtaos.cmd_issueCorrection.start(timeout=60.)
```

[33]:
```
<ddsutil.MTAOS_ackcmd_fd03e870 at 0x7f7fbdca9460>
```

Stop Tracking

[34]:
```
await mtcs.stop_tracking()
```

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

Wrap up. Put each component to the following states: mtaos –> standby m1m3 –> standby m2 –> standby camera hex –> standby m2 hex –> standby

[ ]:
```
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.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()
```