TSOS – Time Based Spacecraft Operation Simulator [1,2]

Wenwen Lu
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grave Dr
Pasadena, CA 91109
818-354-0004
wenwen.lu@jpl.nasa.gov

Abstract—This paper describes the Time Based Spacecraft Operation Simulator (TSOS) system. TSOS is part of the Virtual Mission Operation Framework (VMOF). It is a dynamically configurable executive. TSOS simulates the execution of science observation scenarios on each spacecraft subsystem, propagates subsystem states, and generates the telemetry data. The telemetry data are broadcasted and validated through visualization stations. The capability of TSOS to simulate the execution of a science observation scenario before real operation happens provides a means for early detection of possible design flaws. Thus, it helps reduce mission cost and ensure mission success.

TABLE OF CONTENTS

1. INTRODUCTION

TSOS is part of the Virtual Mission Operation Framework (VMOF) [1], which is an R&D project led by the Mission Simulation and Instrument Modeling Group (MSIM) at the Jet Propulsion Laboratory.

Besides TSOS, VMOF consists of the Model Object Builder (MOB), the Science and Engineering Activity Reasoning and Optimizer (SCENAREO) [2], and the Distributed Telemetry Visualization System (DTVS). In VMOF, MOB provides the subsystem design interface and generates parametric model scripts; SCENAREO is a tool to design science observation sequences based on subsystem properties and science objectives. The model scripts generated by MOB and the observation sequences generated by SCENAREO are fed to TSOS as inputs. TSOS simulates the execution of science experiment scenarios on the virtual prototypes of a spacecraft subsystem. The spacecraft subsystem states are captured in telemetry data. The telemetry data are broadcasted and validated through DTVS visualization stations. Figure 1 shows the system structure of VMOF.

Traditionally, each science team is responsible for the science observation of its one instrument. It is usually during the operation phase when inter-subsystem dependencies on mission system operability are analyzed. If the analysis results are unfavorable, it might cost the reduction of science return, since it is then too late to change the mission system during the operation phase. To avoid such situations, it is important to have a test mechanism of mission system operability during the whole mission lifecycle.

The VMOF system was developed to support the validation-in-the-loop system design process and to support lifecycle-continuous mission operation that can be seamlessly integrated into real operation. TSOS is designed to provide a virtual platform to simulate the execution of science observation scenarios before real operation happens, thus making it possible to achieve early detection of potential design flaws.

This paper is organized as follows: Section 2 describes, the TSOS software architecture; Section 3 describes the execution flow of TSOS. Section 4 provides detailed description of each TSOS component; Section 5 describes DTVS, the Distributed Telemetry Visualization System. The paper is summarized with further work of TSOS.

---

[2] IEEEAC paper #1210

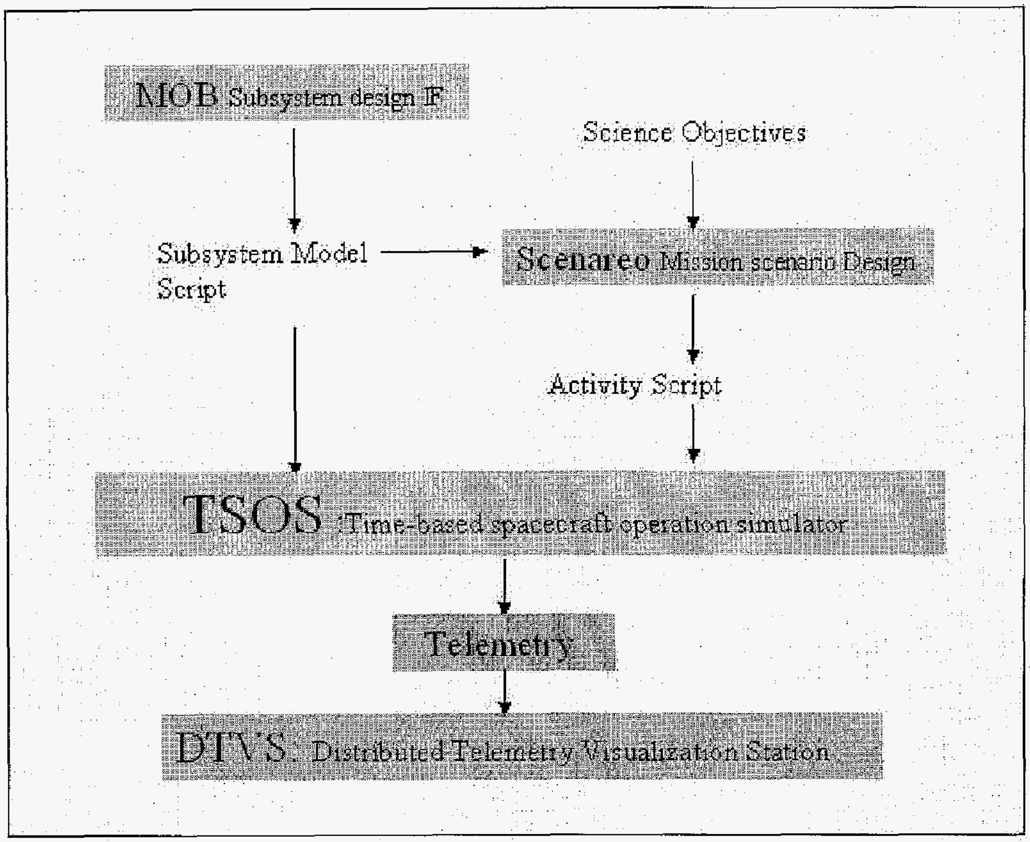


Figure 1. Virtual Mission Operation Framework

2. TSOS ARCHITECTURE

The subsystems in TSOS include Navigation (NAV), Attitude and Articulation Control System (AACS), PAYLOAD, Telecommunication (TELECOM), and POWER. The software architecture of TSOS consists of three modules: Sequence Manager, Virtual Subsystem, and Telemetry, as shown in Figure 2.

The role of Sequence Manager is to parse the science sequence generated from SCENAREO and route the command to the corresponding virtual subsystem. The Virtual Subsystem Module simulates the execution of the command for the corresponding subsystem based on the simulation of the subsystem property. The telemetry module mimics the real-life telemetry. It is a collective data structure for all the subsystem states. The telemetry data generated by TSOS are visualized via DTVS.

The central object of the Virtual Subsystem is V_CDH, which stands for Virtual Command and Data Handling. It has links to each V_Subsystem, where subsystem can be V_AACS, V_NAV, V_PAYLOAD, V_TELECOM, V_POWER. V_CDH is the object that receives commands from the sequence manager and sends those commands to the corresponding V_Subsystem. V_Subsystem executes the commands, propagates the states, and reports the state to V_CDH. V_CDH collects all these states into telemetry data.

Figure 2 shows that V_Subsytem is based on M_subsytem and S_subsystem. Here M_subsystem and S_subsytem are the two major libraries of VMOF. The S_ in S_Subsystem symbolizes Script reader; and the M_ in M_Subsystem symbolizes Model; just as in the convention for the V_ in V_Subsystem, which stands for Virtual Prototype. S_Subsystem is the library that parses subsystem property scripts generated by MOB and creates subsystem property objects. It is a static capture of subsystem property. M_Subsystem, on the other hand, is

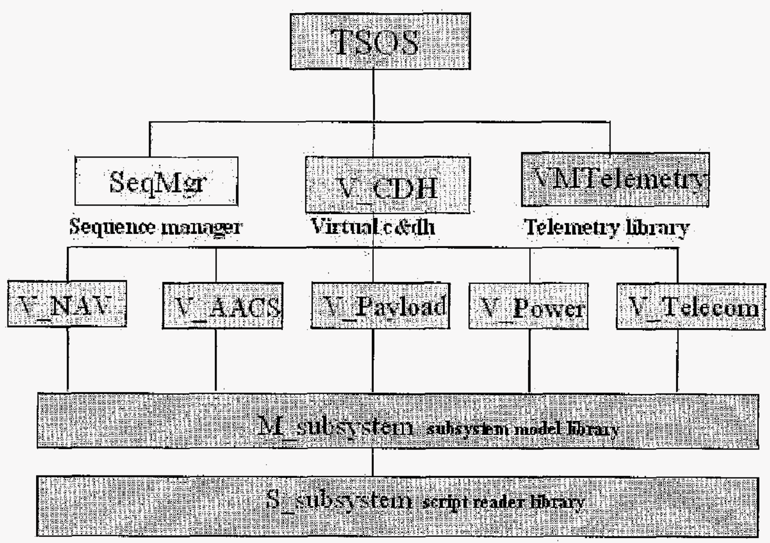


Figure 2. Software architecture of TSOS

derived from S_Subsystem. It is a library that provides functions to compute subsystem performances at various conditions and at various time points. Based on the subsystem performance property modeled by M_Subsystem, V_Subsystem utilizes M_Subsytem and therefore S_Subsytem to simulate the response of each subsystem to different command at different states, and fulfill its goal as a spacecraft simulator.

3. TSOS EXECUTION

As a spacecraft operation simulator, TSOS is designed to have an execution flow that is a strong analogy to the execution of a real mission.

To start the TSOS execution, the first step is initialization. Sequence manger needs to be initialized through the input of a scenario activity script file. Model script files are fed to construct M_CDH, which is the model object of C&DH in M_Subsystem. From here, in M_CDH, all the M_Subsystem objects (the model objects of subsystems) are constructed. A virtual prototype of C&DH and V_CDH, is initialized after M_CDH. All the virtual prototype objects of subsystems, V_Subsystem, are constructed in V_CDH. Therefore, V_CDH is linked to all the V_Subsystem to accomplish its command and data handling. All the virtual prototypes of subsystems, V_Subsystems, can access their model objects, M_Subsystem, to compute the subsystem performance and calculate the subsystem state at various time points under various commands.

For initialization, users also need to provide sampling frequency of telemetry data and output data filename for data recording.

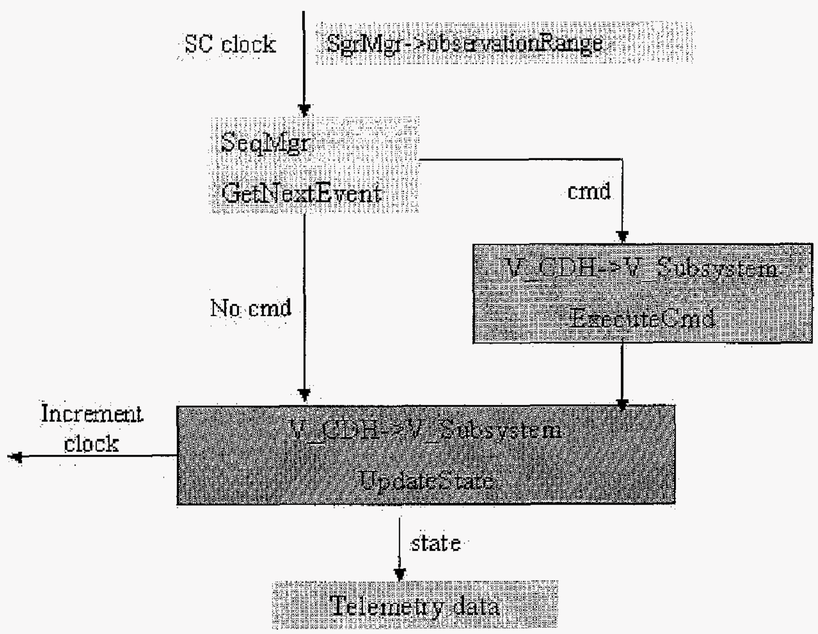


Figure 3. Execution Flow of TSOS

As a timed-based spacecraft operation simulator, TSOS has a virtual spacecraft clock. All the execution is driven by the time-based sequence commands. At each clock step, sequence manager checks to see if the time is in a valid range, e.g., for a chosen orbit, the time point needs to be inside this orbit period. If the time is valid, sequence manager checks to see if there is an event at this time for any subsystem. If there is an event for a particular subsystem, the virtual prototype of C&DH object (V_CDH) receives this command and sends to the corresponding virtual subsystem prototype (V_Subsystem) to simulate the execution of this command, the response of the subsystem to this command is collected by V_CDH. If there is no command to the subsystem, the virtual subsystem prototype, (V_subsystem) propagates its state, and its state again is collected by V_CDH. The collected data are recorded in telemetry format. At the same time, the spacecraft clock is incremented to the next step. This loop continues until all the commands are executed and the valid time range of the spacecraft clock expires.

4. TSOS COMPONENTS

As mentioned above, TSOS has three components: sequence manager, Virtual Prototype of subsystems, and telemetry data. In this section, these three components are described in more details respectively.

4.1 Sequence Manager

Sequence Manager has three functions:

- parse the science scenario script files;
- provide command to subsystem at each time step;
- check if any give time is valid within the observation period.

There are two script files involved here, activity file and observation file. Figure 4 and Figure 5 show examples of these two files generated by SCENAREO. Observation file contains the reference time for each observation. The activity file is a collection of time-sequenced command lists for each subsystem. The time for each command is a relative time to the reference time of this observation saved in the observation file. Sequence manager translates this time-based command list and routes it to the corresponding subsystem at each time step.

4.2 Virtual Prototype Of Subsystem

Virtual prototype of subsystem (V_Subsystem) is the object that receives the command from sequence manager, simulates the responses of the subsystem to the command, and then records the state of the subsystem.

```
S_OBSERVATION
{
  ACTIVITY_DEF =
       "Q:\Missions\NSI\Activity\orbittest.activity"
  OBSERVATION_LIST
  {
    OBSERVATION = "ObsTest-0"
    {
       ORBIT = 0, 0
       ACTIVITY = "EuropaTest-0"
    }
  OBSERVATION = "ObsTest-1"
    {
       ORBIT = 1, 1
       ACTIVITY = "EuropaTest-1"
    }
  }
}
```

Figure 4 An Example of an Observation Script File

V_Subsystem has three operation modes: NEW COMMNAD, IN OPERATION, and READY. When the subsystem receives the command from sequence manager, it is in NEW COMMAND mode. All the command parameters are passed to V_Subsystem and the validity of the command is checked. The initial state and the desired state of the subsystem corresponding to this command are calculated. In the next time step, after the new command is executed, the subsystem is set to the IN OPERATION mode. The subsystem propagates the state of subsystem at each time step and checks if the subsystem has reached its desired state for this command. The subsystem is blocked from receiving new commands under IN OPERATION mode. When the subsystem reaches its desired state, it is switched to READY mode, ready for a new command.

4.3 Telemetry Data

Telemetry data is a collection of data structures of the subsystem states. It mimics the real mission telemetry data. Each data record has a time stamp. Each subsystem has its data structure that contains the parameters of most interest to the mission engineers and scientists. A list of the telemetry data structures for different subsystem is shown in Figure 6.

5. DTVS

Distributed Telemetry Visualization Station (DTVS) is an application of the Micro Helm Project [3]. It is a PC-based mission visualization system that can be employed for comprehensive monitoring of mission system states during real or virtual operation. Micro-Helm has utilized state-of-art hardware and software and developed new hardware and software [?] to achieve its goal. Its distributed visualization capability provides comprehensive monitoring of mission system state. Its scalable PC cluster architecture makes it easy to be replicated and customized for a wide range of missions at different phases of mission lifecycle.

DTVS consists of a cluster of PCs operating in Windows. The CPUs are interconnected with a local area network. Figure 7 shows an example of a DTVS setup with six flat panel monitors concatenated as a 3x2 matrix. Each screen is devoted to visualize one aspect of the mission states.

DTVS has a server/client structure. The communication between server and client are established through socket. The telemetry server first loads the mission models to initialize all the static parameters of all the subsystems in a telemetry header and sends the [parameters?] ___________ to the client. This minimizes the client's computation load. After the telemetry header is sent, the telemetry server packetizes the telemetry record and broadcasts it to the client. Each client visualizes the telemetry data in 2D or 3D. A spacecraft system state can be viewed from different subsystem perspectives simultaneously. Furthermore, multiple states can also be projected into an integrated system state.

Here are a few examples:

- SC View: presents 3D spacecraft attitude, solar panel articulation, antenna articulation, antenna pointing along with the directions of Earth, Sun, and the target body. It also reports telemetry records, such as Earth receiving time and downlink rate (Figure 8).

- Instrument View: presents the view from the instruments, such as the camera view. It shows the target position/size relative to the camera field-of-view. This was proven to be a very useful tool for design and validate sequence for camera instrument during Deep Space 1 (DS1) mission. (Figure 8)

- Footprint View: presents time, a cylindrical map for the target surface with real-time sunlight and shade update, and with footprints projected onto the surface map. Projection of the spacecraft track is also show in this view (Figure 9).

DTVS is a powerful validation tool and a cost-effective system solution. Its flexibility in analyzing the telemetry data in different aspects enables comprehensive understanding of a mission among multi-discipline teams.

6. CONCLUSIONS

Currently, TSOS serves as a science-return validation platform during concept design phase in collaboration with Team-X at JPL. During this collaboration, we expect to improve TSOS to be more mission oriented and more robust. The comprehensive monitoring of a mission during the early design phase is essential for assuring the operational feasibility of a mission and for maximizing the science return.

REFERENCES

[1] Meemong Lee, "Virtual Mission Operation Framework", *IEEE Aerospace Conference, 2004,* Big Sky, Montana ([month 2004]).

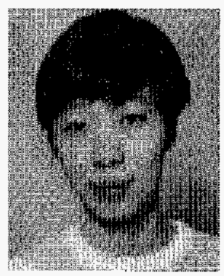
[2] Shin-Ywan Wang, "SCENAREO – Science and Engineering Activity Reasoning and Optimization", *IEEE Aerospace Conference, 2004,* Big Sky, Montana ([month 2004]).

[3] Wenwen Lu, Richard J. Weidner, Meemong Lee, "Micro-Helm, A Mission Operation Visualization System", *5th International Symposium on Reducing the Cost of Spacecraft Ground systems and Operations",* [city and month] *2003.*

ACKNOWLEDGEMENTS

The work described in this paper was performed at Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. The Information Systems Branch of the Research Program Management Division, Next Generation Infrastructure program, Intelligent Synthesis program, and High Performance Communication and Computation program of NASA supported the research.

BIOGRAPHY



Wenwen Lu *is a senior engineer of the Mission Simulation and Instrument Modeling group at Jet Propulsion Laboratory. She has been involved in the Virtual Mission project since she joined the lab in 1997. She leads the Virtual Mission Operation task, developing distributed visualization servers and various analysis software systems for target phenomena modeling and autonomous operation planning. She also supports in-situ Site Characterization activity for the Mars technology program with rock field synthesis and multi-rover-based science instrument operation executive design and development. She has a bachelor's degree from Fudan University in China, and a doctoral degree in high-energy physics from Caltech.*

```
ACTIVITY = "EuropaTest-0"
{
    AACS
    {
        ACS
        {
        }
        SOLARARRAY = "PrimaryPanel"
        {
                 822 RESET;
                 823 TRACK SUN;
                8363 TRACK_OFF;
        }
        ANTENNA = "HGA"
        {
                 822 RESET;
                 823 SET_RATE 0.054063 0.054063;
                 824 POINT EARTH;
                3829 TRACK EARTH;
        }
    }
    PAYLOAD
    {
        CAMERA = "NAC"
        {
                 822 RESET;
                3180 SET_RATE 1.333333 1.333333;
                5538 POINT EUROPA -1.130000 346.770000;
                5571 TRACK EUROPA -1.130000 346.770000;
                5573 EXPOSE 0.060000 "NAC-0-1.dat";
                5575 READ_ALL "NAC-0-1.dat";
                5576 SET_RATE 1.333333 1.333333;
                5577 TURN 0.0 0.0;
                6457 SET_RATE 1.333333 1.333333;
                7304 POINT EUROPA 74.600000 236.450000;
                7588 TRACK EUROPA 74.600000 236.450000;
                7590 EXPOSE 0.060000 "NAC-0-2.dat";
                7592 READ_ALL "NAC-0-2.dat";
                7593 SET_RATE 1.333333 1.333333;
                7594 TURN 0.0 0.0;
        }
        RADAR = "Radar"
        {
                 823 ON "Radar-0.dat";
                3839 OFF;
        }
    }
    TELECOM
    {
            823 ANTENNA HGA;
            824 SET DOWNLINK 211892.838833;
            3839 DOWNLINK "Radar-0.dat" 80.000000;
            4295 SET DOWNLINK 211850.389139;
            5576 DOWNLINK "NAC-0-1.dat" 10.000000;
            7593 DOWNLINK "NAC-0-2.dat" 10.000000;
    }
    POWER
    {
        HGA
        {
            3839 SET "DOWNLINK" 180.000000;
            5576 SET "DOWNLINK" 10.000000;
            7593 SET "DOWNLINK" 10.000000;
        }
        NAC
        {
            5571 SET "TRACK" 2.000000;
            5573 SET "EXPOSURE" 1.000000;
            7588 SET "TRACK" 2.000000;
            7590 SET "EXPOSURE" 1.000000;
        }
        SOLARARRAY
        {
             823 SET "TRACK" 7540.000000;
        }
        RADAR
        {
             823 SET "ON" 3016.000000;
        }
    }
}
```

Figure 5. An example of an Activity Script File.

<u>TelemNav</u>
ScPosistion
ScVelocity

<u>TelemAntenna</u>
AntennaName
TargetName
IntersecLoc

<u>TelemSoloar Array</u>
IncidenceAngle

<u>VmGimbalData</u>
AngleNo
GimbalAngle
PointingDir

<u>TelemTelecom</u>
AntennaName
FileName
DownlinkRate
DataVolume
Unit

<u>TelemInstrument</u>
InstrumentName
TargetName
IntersecLoc
FileName
CurrentFileSize
TotalDataVolume
CurrentDataRate
GimbalData

<u>TelemPower</u>
TotalLoad
MoreFollowed
PowerShunt
PowerLoad
PowerBattery
PowerSource
SocMinAmp
SocAmp
VolBatt
VolBus
CurrentOrbit
StartTime
EndTime
SunAngle
SunDistance

Figure 6. Telemetry Data Structure

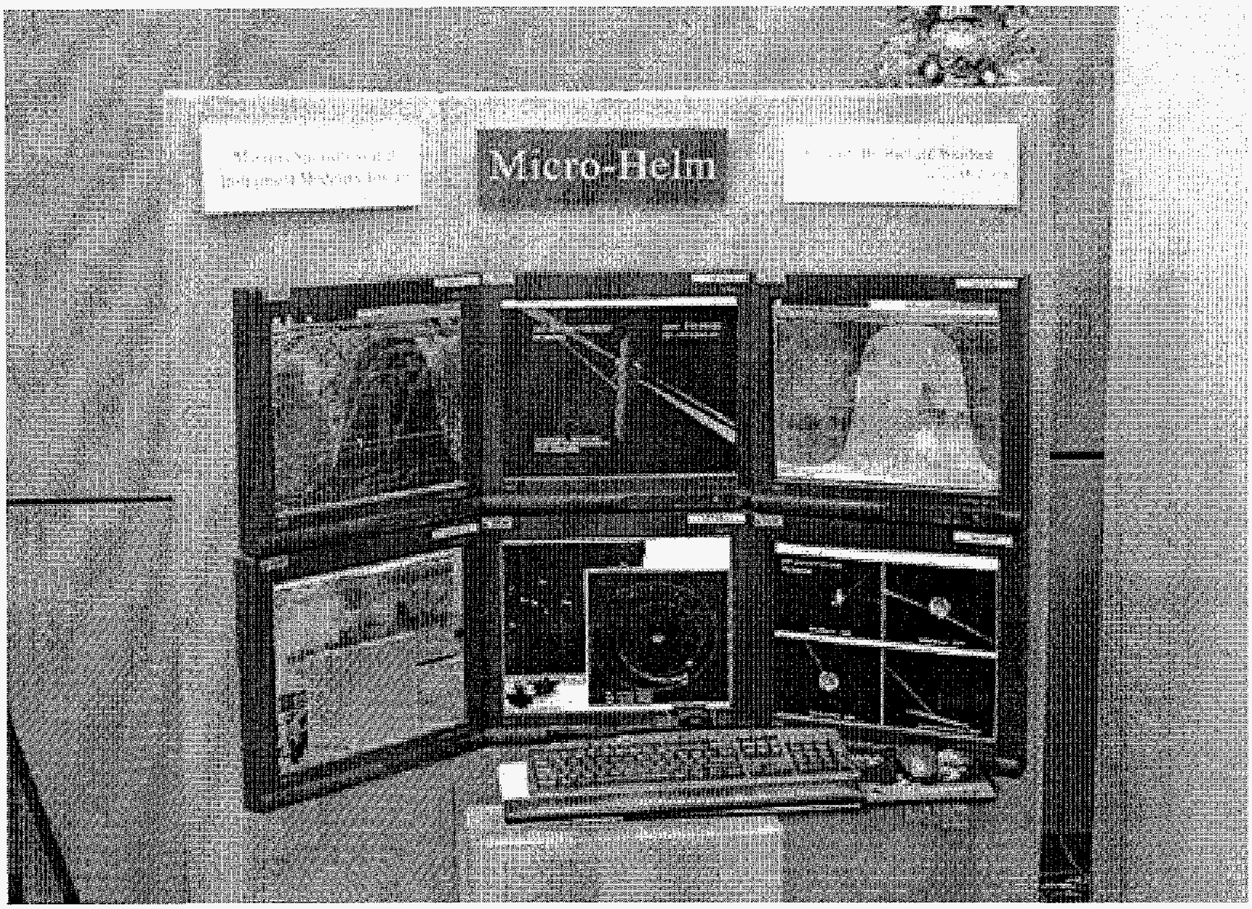


Figure 7. A typical Micro-Helm Display Configuration.

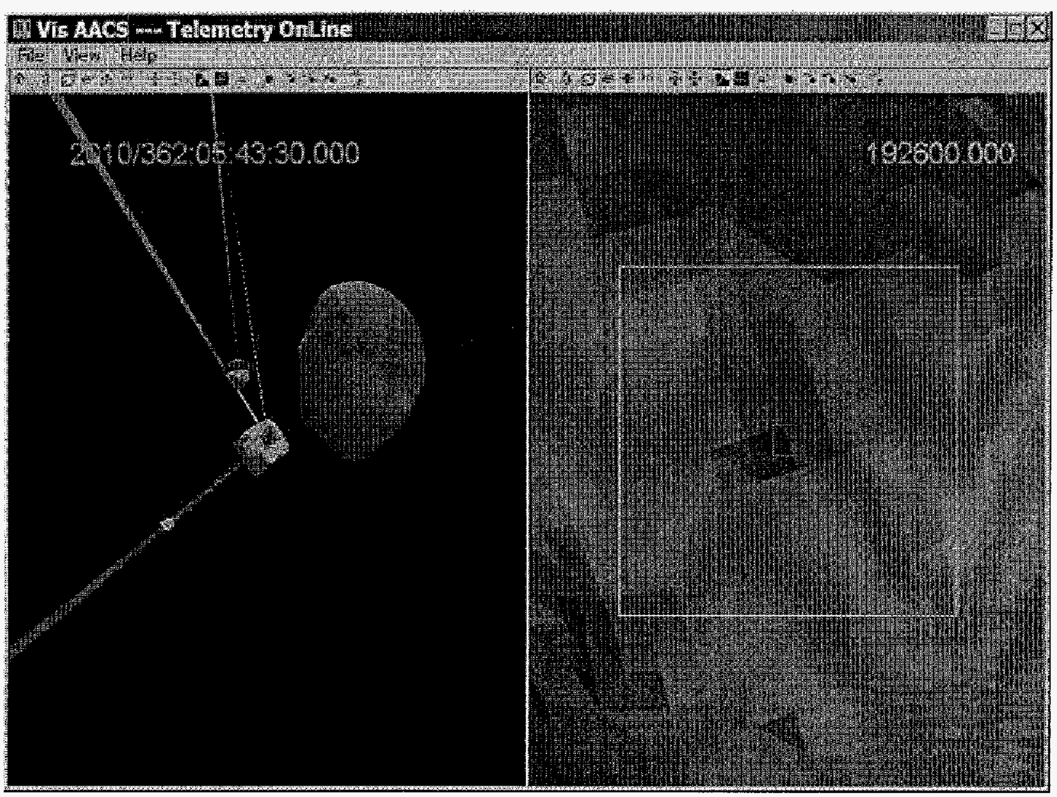


Figure 8. Micro-Helm Spacecraft View (left) and Instrument view (right).