```
mission mbts_sci2 {
  '''
  This mission is basically Science/sci2 with 5 pre-established
  waypoints in the order: C1 - M1 - M2 - M1 - C1, but also with ability
  to (re)start the mission at any point within this trajectory, just
  adjusting the `trajectory` parameter as needed.
  '''

  arguments {
    # Almost every mission should start with an overall timeout and a NeedCommsTime.

    MissionTimeout = 2 hour
    '''
    Maximum duration of mission
    '''

    NeedCommsTime = 60 minute
    '''
    How often to surface for communications
    '''

    # You probably need to change these.

    trajectory = 1 count
    '''
    Denotes the desired concrete MBTS trajectory. By default, this is 1
    meaning the complete trajectory: C1 - M1 - M2 - M1 - C1.
    Valid values are 1–5, with the following effect:
        1: C1 - M1 - M2 - M1 - C1;
        2: M1 - M2 - M1 - C1;
        3: M2 - M1 - C1;
        4: M1 - C1;
        5: C1.
    '''

    # You probably do not need to change these.

    Lat[1..5] = [36.797 degree, 36.75 degree, 36.691 degree, 36.75 degree, 36.797 degree]
    '''Do not need to change this (used internally). Latitude of the 5 waypoints.'''

    Lon[1..5] = [-121.847 degree, -122.022 degree, -122.376 degree, -122.022 degree, -121.847 degree]
    '''Do not need to change this (used internally). Longitude of the 5 waypoints.'''

    Speed = 1 meter_per_second
    '''
    Speed while performing the YoYo behavior.
    '''

    CaptureRadius = NaN meter
    '''
    Speed while performing the YoYo behavior.
    '''

    YoYoMinDepth = 2 meter
    '''
    Minimum depth while performing the YoYo behavior.
    '''

    YoYoMaxDepth = 200 meter
    '''
    Maximum depth while performing the YoYo behavior.
    '''

    YoYoMinAltitude = 9 meter
    '''
    Minimum altitude while performing the YoYo behavior (for
    bottom-terminated YoYos).
    '''

    YoYoUpPitch = 20 degree
```

```
    '''
    Vehicle up pitch while performing the YoYo behavior.
    '''

    YoYoDownPitch = -20 degree
    '''
    Vehicle down pitch while performing the YoYo behavior.
    '''

    # You are even less likely to need to change these.

    BuoyancyNeutral = Control:VerticalControl.buoyancyNeutral
    '''
    Buoyancy bladder position while performing the YoYo behavior. Defaults
    to buoyancyNeutral setting in the Config/Control.cfg file. Set to NaN cc
    for active buoyancy
    '''

    MassDefault = Control:VerticalControl.massDefault
    '''
    Static setting for mass during the mission. Set to NaN mm for active
    mass position
    '''

    MinAltitude = 5 meter
    '''
    Minimum height above the sea floor for the entire mission.
    '''

    MaxDepth = 205 meter
    '''
    Maximum depth for the entire mission.
    '''

    MinOffshore = 2 kilometer
    '''
    Minimum offshore distance for the entire mission.
    '''
}

# Missions should almost always start with a timeout

timeout duration=MissionTimeout

insert "Insert/Science" {
    '''
    Most missions will run the science sensors. If you don't place this
    aggregate above NeedComms, science instruments get turned off on the
    last upcast and while floating on the surface.
    '''
    redefineArg SampleISUS = true

    redefineArg PeakDetectChlActive = true
}

# Most missions should use the NeedComms aggregate.
insert "Insert/NeedComms" id="NeedComms"

assign in sequence NeedComms:DiveInterval = NeedCommsTime

# Missions should almost always start with standard safety envelopes;
# most missions should not expose the parameters of these envelopes.

insert "Insert/StandardEnvelopes"

assign in sequence {
    StandardEnvelopes:MinAltitude = MinAltitude
    StandardEnvelopes:MaxDepth = MaxDepth
    StandardEnvelopes:MinOffshore = MinOffshore
}
```

```
141
142   assign in sequence StandardEnvelopes:MaxDepth = MaxDepth
143
144   assign in sequence StandardEnvelopes:MinOffshore = MinOffshore
145
146   call id="StartingMission" refId="NeedComms"
147
148   # Many missions will keep mass position and buoyancy volume fixed at defaults.
149
150   behavior Guidance:Pitch in parallel {
151     setting massPosition = MassDefault
152   }
153
154   behavior Guidance:Buoyancy in parallel {
155     setting position = BuoyancyNeutral
156   }
157
158   behavior Guidance:SetSpeed in parallel {
159     setting speed = Speed
160   }
161
162   behavior Guidance:DepthEnvelope in parallel {
163     '''
164     Another depth envelope for the YoYo behavior. This envelope should
165     fall within the limits of the standard safety envelopes in
166     Insert/StandardEnvelopes.xml in order to avoid commanding high pitch
167     angles for depth-terminated YoYos.
168     '''
169     setting minDepth = YoYoMinDepth
170     setting maxDepth = YoYoMaxDepth
171     setting downPitch = YoYoDownPitch
172     setting upPitch = YoYoUpPitch
173   }
174
175   behavior Guidance:AltitudeEnvelope in parallel {
176     '''
177     Another altitude envelope for the YoYo behavior. This envelope
178     should fall within the limits of the standard safety envelopes in
179     Insert/StandardEnvelopes.xml in order to avoid commanding high pitch
180     angles for bottom-terminated YoYos.
181     '''
182     setting minAltitude = YoYoMinAltitude
183     setting upPitch = YoYoUpPitch
184   }
185
186   behavior Guidance:YoYo in parallel {
187     setting downPitch = YoYoDownPitch
188     setting upPitch = YoYoUpPitch
189   }
190
191   aggregate SetWaypointsPerTrajectoryParameter {
192     '''
193     The following Trajectory2-5 aggregates adjust the Lat[] and Lon[] variables
194     according to the `trajectory` parameter:
195     '''
196     run in sequence
197
198     aggregate Trajectory2 {
199       run in sequence
200       break if ( trajectory <= 1 count )
201       assign in sequence {
202         Lat[1] = NaN degree
203         Lon[1] = NaN degree
204       }
205     }
206
207     aggregate Trajectory3 {
208       run in sequence
209       break if ( trajectory <= 2 count )
210       assign in sequence {
```

```
        Lat[2] = NaN degree
        Lon[2] = NaN degree
      }
    }
    aggregate Trajectory4 {
      run in sequence
      break if ( trajectory <= 3 count )
      assign in sequence {
        Lat[3] = NaN degree
        Lon[3] = NaN degree
      }
    }
    aggregate Trajectory5 {
      run in sequence
      break if ( trajectory <= 4 count )
      assign in sequence {
        Lat[4] = NaN degree
        Lon[4] = NaN degree
      }
    }
  }

  aggregate Lap {
    run in sequence

    syslog important "Starting Lap"
    syslog important "Lat[1] = " + Lat[1] + " Lon[1] = " + Lon[1]
    syslog important "Lat[2] = " + Lat[2] + " Lon[2] = " + Lon[2]
    syslog important "Lat[3] = " + Lat[3] + " Lon[3] = " + Lon[3]
    syslog important "Lat[4] = " + Lat[4] + " Lon[4] = " + Lon[4]
    syslog important "Lat[5] = " + Lat[5] + " Lon[5] = " + Lon[5]

    # The following aggregates dispatch the Guidance:Waypoint's as usual
    # (similarly as with sci2 but with up to 5 waypoints here):
    macro $i = 1..5 {
      aggregate Wpt$i {
        run in sequence

        behavior Guidance:Waypoint in sequence {
          setting latitude = Lat[$i]
          setting longitude = Lon[$i]
          # setting captureRadius = CaptureRadius
        }
      }
    }
  }
}
```