








No.	C. Android Remote Controller Module Functional Specifications	MDP Supervisor Signature / Date
C.1	<p><b>The Android application (AA) is able to transmit and receive text strings over the Bluetooth serial communication link.</b></p> <p>Note: You can use the AMD tool to help verify that your AA has successfully achieved bi-directional data transfer.</p>	
C.2	<p><b>Functional graphical user interface (GUI) that is able to initiate the scanning, selection and connection with a Bluetooth device.</b></p> <p>E.g. when the Connect button is touched, a list of available devices is presented to the user for selection. Once a device is selected, a connection is established with the device. You can use C.1 to show evidence of a successful connection.</p>	
C.3	<p><b>Functional GUI that provides interactive control of the robot movement via the Bluetooth link</b> (e.g. move forward, left and right). The interactive control of the robot movement can be done using several labeled buttons (minimal requirement), appropriate touch gestures, button cum device tilt or any other method you can think of. You can use the AMD tool to demonstrate control of the virtual robot movement.</p> <p>Caution: Manually entering different string commands in a text box to control the robot movement is not a valid implementation of this requirement.</p>	
C.4	<p><b>Functional GUI that shows remote update &amp; status messages</b> (e.g. ready to start, looking for target 2, etc). You can implement this using a TextView box (minimal requirement). You can use the AMD tool to simulate information update by devising your own string-based protocol representing the various possible status of your robot.</p> <p>Note: Your TextView box must only display selective information and not all the text data that is being streamed to Android tablet.</p>	
C.5	<p><b>2D display of the exploration arena with obstacles and the robot's location.</b></p> <p>E.g. you can create a drawing canvas on your GUI where square numbered obstacle blocks (from 1, 2, 3,..n) can be drawn within a bounded exploration arena. The number text drawn inside your obstacle should be in small white colored fonts and it represents your assigned number to each new obstacle added to your map</p> <div data-bbox="331 1451 505 1520" data-label="Image"> </div> <p>(e.g.  or ). A robot is also drawn at a specified coordinate (x,y) and its facing direction (N,S,E,W) can be clearly inferred from the robot icon displayed.</p>	
C.6	<p><b>Interactive movement and placement of obstacles in map.</b></p> <p>Your GUI must allow you to interactively place the square obstacles into the map through touch interactions on your map area. You must also allow these obstacles in the map to be moved around within the map through a “touch and drag” interaction. Dragging the obstacle outside the map area will remove the obstacle from your map. Once the positioning of the obstacle is completed and the finger is lifted, the (x,y) coordinates and number assigned to the obstacle is transmitted out via the Bluetooth channel. You are free to devise the string format for this information.</p>	

**CE/CZ3004 - Multi-disciplinary Design Project (MDP)**  
**(Assessment Component)**

C.7	<p><b>Interactive annotation of the face of the obstacle where the target image is located.</b> Your GUI must provide functionality that will allow you to indicate which of the side of any particular obstacle touched has the target image. If your obstacles are too small to touch one of the four sides, device another method that will allow you to do this task. Any alternative method to specify which side of the four faces has the target image must still be a touch-based interaction. Once the target face has been registered, the appearance of the obstacle must</p> <p>change to indicate which obstacle face has the target image (e.g. ) and the target face and obstacle coordinate must be communicated via the Bluetooth channel. You are free to devise the string format for this information.</p>	
C.8	<p><b>Robust connectivity with Bluetooth device.</b></p> <p>Your Android application (AA) must not hang up if connectivity with the Bluetooth device is temporarily lost (e.g. by executing a Disconnect at the AMD tool after connection has been established). Your AA should automatically re-established connection automatically once the Bluetooth device connects with the AA again (e.g. by executing a Connect again at the AMD tool after connection was earlier broken with a Disconnect).</p>	
C.9	<p><b>Displaying Image Target ID on Obstacle Blocks in the Map.</b></p> <p>The appearance of any numbered obstacle block can be changed to display a Target ID (in large white fonts) when the Bluetooth channel receives the string "TARGET, &lt;Obstacle Number&gt;, &lt;Target ID&gt;". For example, the obstacle block appearance changes to , , etc. The face in which the target image is located is displayed with a thick visible line of a distinguishing color (e.g.  or )</p>	
C.10	<p><b>Updating Position and Facing Direction of Robot in the Map.</b></p> <p>The position of the robot and the direction the robot is facing can be updated in the map of your Android tablet when the Bluetooth channel receives the string "ROBOT, &lt;x&gt;, &lt;y&gt;, &lt;direction&gt;", where &lt;x&gt; and &lt;y&gt; are valid integer coordinates in your map and &lt;direction&gt; is any one of four directions (N, S, E, W).</p>	