RAPID PROTOTYPING AND EVALUATION OF AUTOMOTIVE HMI IN VIRTUAL ENVIRONMENT

VTTI-00-026: Guiding Driver Responses during Manual Takeovers from Automated Vehicles

Principal Investigator: **Hyungil Kim**

Prepared by: **Alexander Krasner**

Date: 6/1/2020

OUTLINE

- Project Overview
- The Simulator
- The HMIs
- Scenarios and Data Logging
- Examples
- Takeaways

PROJECT OVERVIEW

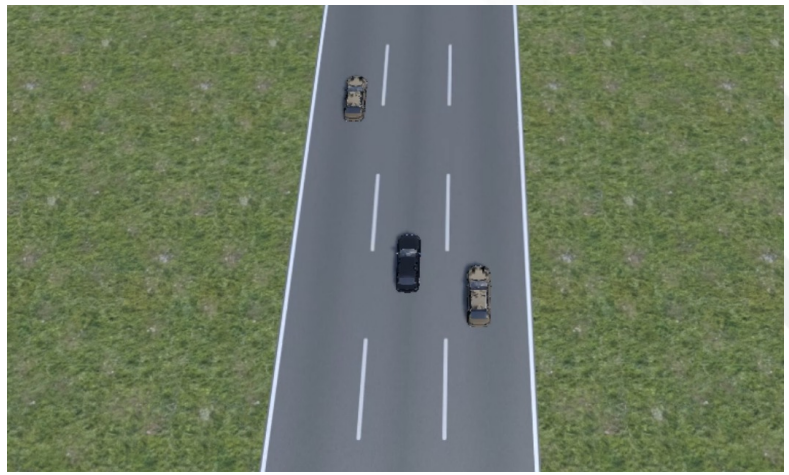
- As self-driving cars improve, there will be limits to capability
- When system is unable to form correct response, issues a “takeover request”
  - Request for user to take control of vehicle
- What and how should we communicate information during takeover to inform user decision?
  - Users need to react quickly and correctly

PROJECT GOALS

- Test several Human Machine Interfaces (HMI) as ways to communicate takeover information
  - Steering wheel haptic feedback
  - Audio messages
  - Augmented reality overlay
- How will these affect how users react?

THE SIMULATOR

- Subject placed in virtual car on straight road
- Driver enables autopilot and the car drives until takeover request is issued
- Secondary visual search task is provided
- Subject responds to situation and takes action



THE SYSTEM

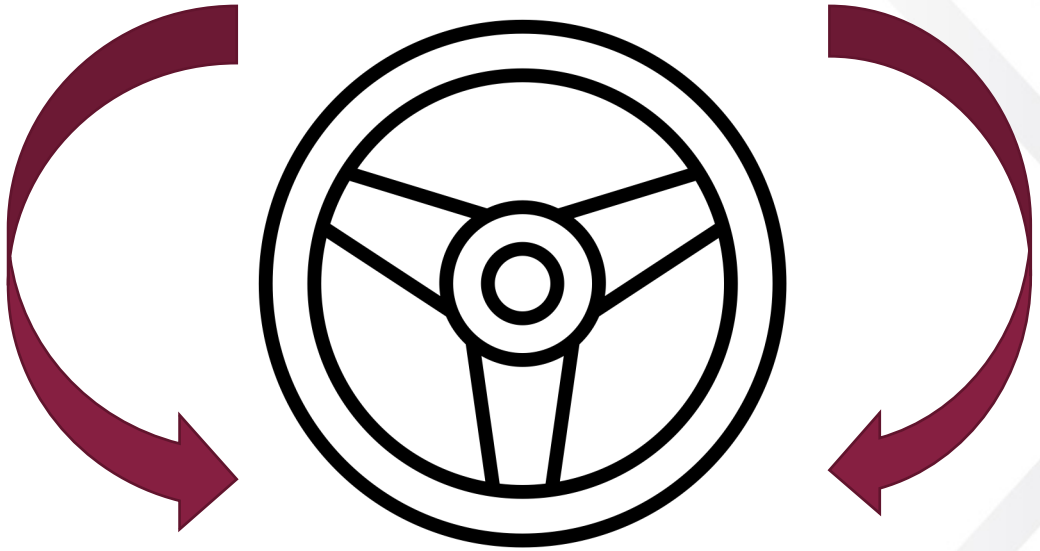
- Visual Display
  - Unity Engine
  - HTC Vive with Tobii Pro Eyetracking

- Steering control and haptics
  - Logitech G920 wheel and pedals

https://www.tobiipro.com/product-listing/vr-integration/

WHEEL HMI

- Three forms tested
  - No reaction
  - Wheel Nudge
  - Turn Resist
- Nudge
  - Wheel would jerk lightly in the direction that is known to be safe to turn
- Resist
  - Wheel would give some resistance to turning in the direction of an adjacent obstacle



WHEEL IMPLEMENTATION

- Nudge is a simple collision effect played from the Logitech Steering Wheel SDK
- Resist was a tougher problem to solve
  - Users overpower the wheel to the point that even at peak power, the resistance can go unnoticed
  - Need to detect when a turn would send the car into an obstacle rather than when a turn is in the direction of the obstacle
    - Example: user is moving diagonally to the left to change lanes, they will need to turn slightly right to straighten out in lane

WHEEL IMPLEMENTATION CONT.

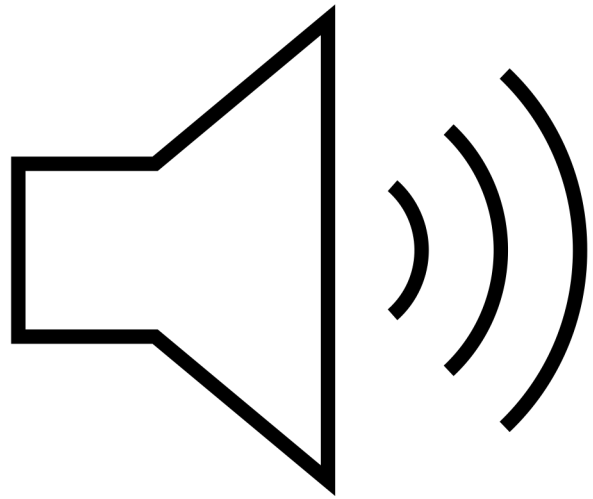
- Solution
  - Dynamically shift how far to the wrong direction a subject must turn before they feel resistance
    - Scale the required turn angle to trigger the resistance with the velocity of the car in the correct direction
    - This also means that for all significantly incorrect wheel positions, resistance will be felt if turning the wrong direction
      - For example, if the wheel is not turned at all but the car is positioned to move towards the wrong lane, that is a wrong wheel position
  - Make resistance noticeable by rapidly pulsing the effect

NUDGE AND RESIST DETAIL

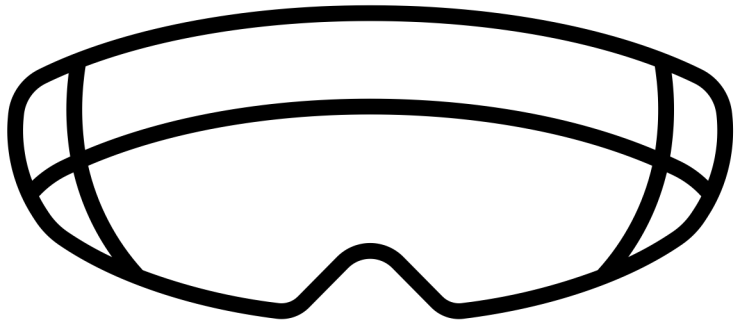
- Implemented using Logitech Steering Wheel SDK
  - https://www.logitechg.com/en-us/partnerdeveloperlab/sdk-resource-list/steering-wheel-sdk.html
- Nudge uses a LogiPlaySideCollision at 60% Power
- Resist uses a variable Power of LogiPlayConstantForce
  - When the user turns the wheel past a certain angle in the wrong direction, the constant force is activated, pulsing on/off every 0.2 seconds
  - As the velocity of the vehicle increases towards the wrong direction, the angle of activation is slightly shifted more towards the correct turning direction
    - This makes sure that the wheel will “resist” driving with a centered wheel with the car oriented diagonally in lane towards the wrong direction

AUDIO HMI

- Three forms tested
  - Warning Beep
  - Beep followed by “Takeover Required”
  - Beep followed by “Turn Left/Right”
- Simple implementation
  - Audio clips saved and played in Unity

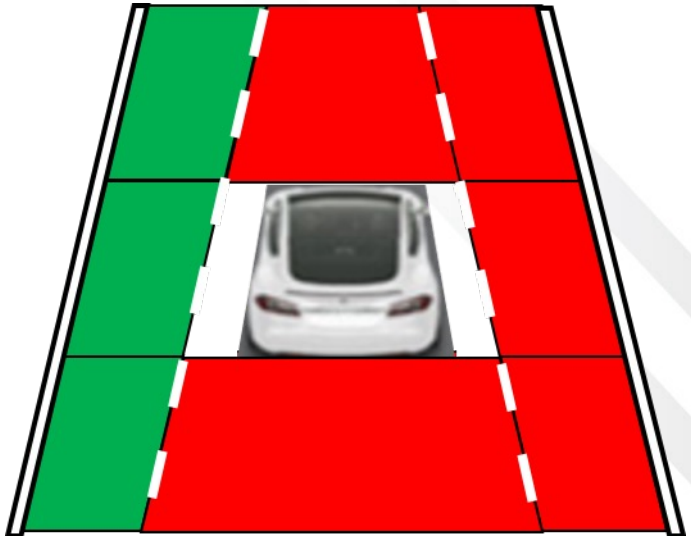


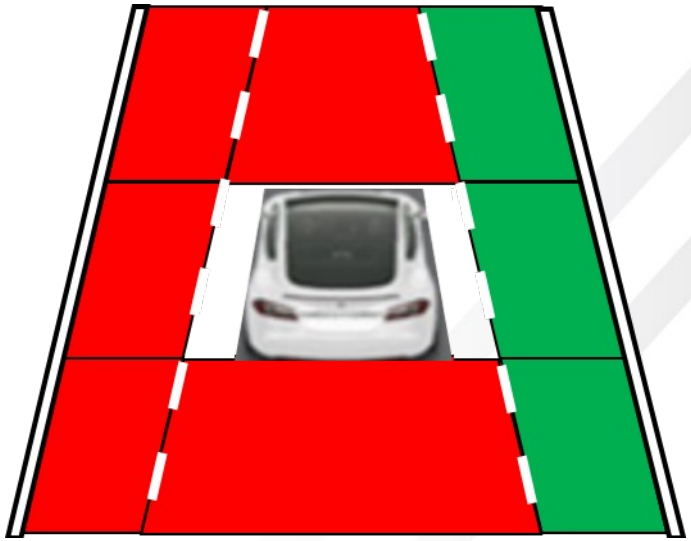
VISUAL HMI



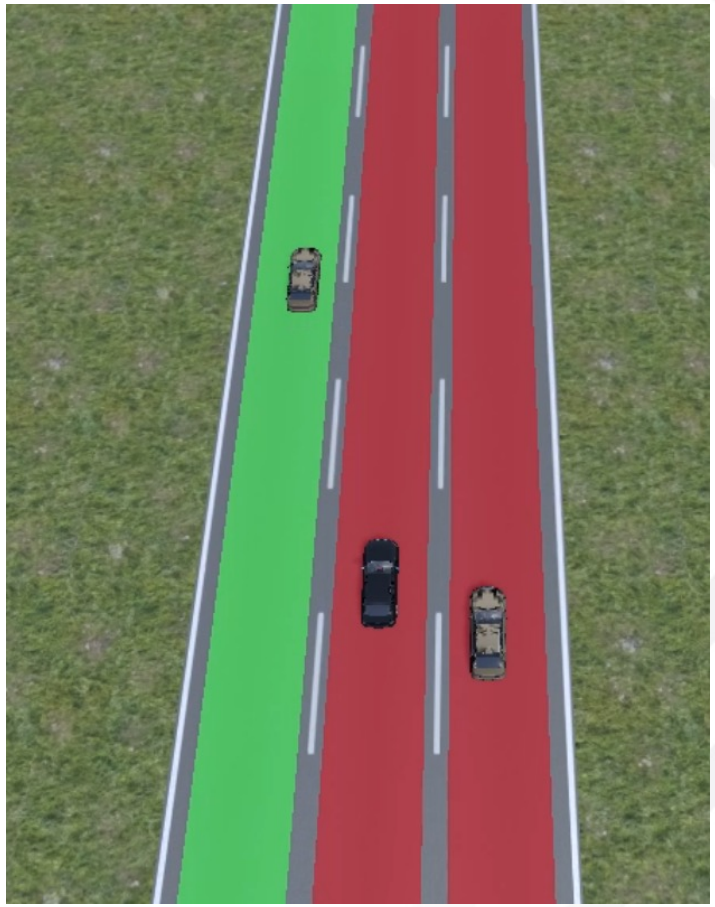
- Three forms tested
  - No effect
  - Screen-Fixed AR Icon
  - World-Fixed AR Lane Colors
- Screen-Fixed
  - Displayed a static icon showing the car and colored lanes around it
- World-Fixed
  - Colored the lanes in the virtual world to indicate what lane is safe
- Same color system used between both types
  - Examples seen in next slides

SCREEN-FIXED EXAMPLE





WORLD-FIXED EXAMPLE



CONSTANT HMI

- Gauge cluster autopilot icon
  - Would display status of autopilot
  - On, off, or disabled

- Center Console Takeover Icon
  - Replaced the secondary task on the center console during takeover

CREATING SCENARIOS

- All trials were run as scenarios detailed in an XML file
- A script was run to generate the trials for each participant based off of a detailed, counterbalanced, excel file of all specifications for each trial and participant
- In each set of trials, additional practice scenarios were added at the beginning

SCENARIO VARIABLES

- Several variables of interest
  - Obstacle Type
    - The type of obstacle that the user would crash into if they did not turn
      - Stopped car
      - Traffic cones
      - Cardboard box
  - Distractor Type
    - What kind of action was going on around the car to distract from the primary obstacle
      - Car ahead accelerates
      - Car ahead changes lanes
      - Red car ahead
  - In all scenarios a car was placed in one of the side blind-spots and behind the participant's vehicle

OTHER PROBLEMS OF INTEREST

- Two variables for the specifications in all trials were determined through testing
  - Takeover Alert Trigger Delay
    - The time before colliding with the obstacle at which an alert would be issued
    - Found that 5 seconds was enough time to react without being too far in advance
  - Autopilot Disable Delay
    - The time after the alert is triggered at which autopilot is disabled and the car is locked into manual driving
    - Found 2 seconds to be a reliable time that also fit with previous standards

DATA LOGGING

- Many kinds of data logged by the sim
  - Video
    - Bird's eye and First-person
  - Eye tracking data
    - What objects were they looking at and when
  - Status of all input devices
  - Positions of vehicles
  - Voice recording during and between trials
    - Using headset microphone

VIDEO EXAMPLES

DATA REDUCTION

- Data from trials was reduced for analysis using MATLAB
- Processed data to extract only window of data after alerts were presented
- Reaction times found by looking for first instances of participant input in window
- Minimum Time-To-Collision found by finding lowest value in window
- Eye gaze data processed into two types of results
  - Dwell – how long eyes looked at each object during event window
  - First and Second Glance – finding respective times the object is glanced at during the window

TAKEAWAYS AND FINDINGS

- Though it is not a research project in itself, information can be gleaned from the simulator design process

- Found that:
  - Constant force-based resistance goes undetected and therefore should be pulsed to emphasize the forces to the user
  - As seen in prior work, AR interfaces can be adequately prototyped and tested in VR

LESSONS LEARNED

- Full pilot test runs are critical for testing experiment code
  - No matter how perfectly something works in testing, there can always be issues that you would never have foreseen
- When stuck working on old code, a lot of time can be saved by speaking with those who have already worked on it
- Note for future developers: the Logitech steering wheel system is very finicky
  - Will only provide proper turning feedback if a frontal collision effect is played after each reboot
  - Random reboots and disconnects can occur if let sit too long, build in ways to test for connection

THANK YOU!