# OpenLAP Documentation

The creator of OpenLAP, Michael Chalkiopoulos, created several videos explaining OpenLAP. The six-part series dives into the inner workings of each of OpenLAP’s files. After watching each of the videos, I have created a summary of the videos.

Source: [Michael Chalkiopoulos - YouTube](https://www.youtube.com/@MrHalkiopoulos)

GitHub: [https://github.com/mc12027](https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqa3prYjEwTW9nZ0hXYk9jWGNKNzI4TEozNUV5d3xBQ3Jtc0ttVDhWSkx2dGVUMHBDUmtONkpOZU9seGZ3enRhMFpLcjhrd0pkbzY5SHpFUDREcjZBZ0h1Q054UFduUjJRYW0zRHBMbWhqLWIyWE9iX2lNNDh6ei05SFFfZ3g0T1BRUmNpekRZMkIzMHBVMHR1TjlVSQ&q=https%3A%2F%2Fgithub.com%2Fmc12027&v=WWaouT6EhJ0)

# Summarizing the Videos

## Part I: Introduction

What is OpenLAP?

* Lap time Simulation Software
* Coded in MATLAB
* **Open Source**
* GNU General Public License v3.0

What is lap time simulation?

* Tool for estimating
  + Lap time
  + Response
  + Performance
  + Loads
  + Etc.
* Key point: Driving car at “grip limit”

OpenLAP Project details

* Point Mass approach
* Distance-based solver
* Capabilities
  + Create vehicles
  + Create circuits
  + Simulate steady state lap
  + Simulate straight line acceleration and braking

Vehicles in OpenVEHICLE

* Point mass modeling
* Includes:
  + Tires
  + Engine
  + Drivetrain
  + Aerodynamics
* Generated from tables in excel

Circuits in OpenTRACK

* Includes:
  + Type
    - Closed & Open
    - Direction & Mirroring
  + Curvature / Shape
  + Elevation
  + Banking
  + Localized grip
  + Sectors
* Generated from:
  + Segments
  + Logged data

Straight line simulation in OpenDRAG

* Simple straight line simulation
* Time-based solver
* Longitudinal vehicle model
* Engine torque curve
* Gear shift duration
* Speed traps
* Reports

## Part II: Lap Time Simulation Algorithm

Steady-State VS Transient

* Steady State
  + States = time independent
  + Can solve only for a settled vehicle state
  + Can solve apexes independently
  + Basic equations of motion
* Transient
  + States = time dependent
  + Can solve any vehicle state
  + Cannot solve apexes independently
  + Really complicated

The simplest vehicle model

* Logged data scatter XY plot:
  + X axis = lateral G
  + Y axis = longitudinal G

The simplest track model

* Every point on the track has a turning radius value
  + Straight: R = inf
  + Corner: R = real number

Steady-State LTS algorithm

* Goal 🡪 Calculate velocity
* Steps:
  + Find apex points and speed values
  + Accelerate
  + Decelerate
  + Braking points at intersection

Steady-State LTS example

* Vehicle GG plot
  + A screenshot of a video game

    Description automatically generated with medium confidence
* Track: Simple oval
  + A picture containing diagram

    Description automatically generated
* Steps:
  + Mesh track
    - Lets use 50m segments
  + Locate apexes
    - Due to simplicity, apex will be at radius local minima
  + Find apex speeds
  + Accelerate from all apexes
  + Decelerate from all apexes
  + Get final speed curve
  + Calculate lap time

Final Solution

* Laps oval in 26.502[s]
* What did we learn?
  + A vehicle and track model is needed
  + We need columns to calculate speed from each apex.
  + For closed track, we need to loop back to the first or last point
* What happens when the GGV map changes?
* Speeding things up:
  + Check for overshooting
  + Ignore points with slower preexisting solutions
  + Sort apexes in terms of apex speed

## Part III: Vehicle modeling in OpenVEHICLE

Vehicle Generation Workflow

* Enter excel input
  + Vehicle Data
* Send through OpenVEHICLE
* Get .mat output file
  + Braking model
  + Steering model
  + Driveline model
  + Shift points
  + Force model
  + GGV map

Axes convention

* Right hand rule
* Positive direction
  + X = forward
  + Y = left
  + Z = up
  + θ (Inclination) = downhill
  + ϕ (Banking) = favors right turns
  + ω (Yaw) = left turn

External Forces

* Z Axis
* X Axis
* Y Axis
* Diagram

  Description automatically generated with medium confidence

Tire forces

* General Formula
* Longitudinal acceleration
* Lateral Acceleration
* Combined

Final equations of motion

* Velocity
* Distance
* Circular motion
* Acceleration (Newtons 2nd Law)

Time elimination 🡪 Distance solver

* Velocity
* Distance
* Rearrange velocity equation and substitute into distance
* Expand and substitute
* Combine like terms

Brake model

* Master cylinder area
* Caliper area
* Brake model
* Brake pressure
* Brake pedal force

Steering model

* Diagram, schematic

  Description automatically generated
* Assumptions
  + Small angles
  + No induced drag
* Linear Tire forces and yaw moments
* Target values
* Linear system
* Steering wheel angle

Powertrain model

* Wheel speed and torque
* Vehicle speed
* Maximum tractive force
* Fuel consumption

Driver throttle and brake inputs

* Assumption
  + Linear torque vs tps map
* Acceleration
* Deceleration
* Limits

Optimizing engine tractive force via gear selection (For Loop)

* For % loop through all velocity values
  + For % loop through all gears
    - Back calculate
    - For each value of , calculate:
      * Engine torque
      * Tractive forces
  + End
  + Select gear:
  + Calculate final tractive force curve:
* End

Gearing model example

* Chart

  Description automatically generated

Traction model example

* Chart, line chart

  Description automatically generated

Friction ellipse usage example

* Problem: find the maximum throttle input at a specific speed and track location
* Solution:
  + Calculate external, tire, and engine tractive forces
  + Calculate lateral acceleration that is being used:
  + Calculate remaining longitudinal acceleration from the tires
    - If then the grip isn’t enough and complex numbers appear
  + Calculate throttle value:

GGV map visualization (no drag)

* Chart, funnel chart

  Description automatically generated
* A picture containing text, electronics

  Description automatically generated
  + Green and blue ellipse:
    - Friction ellipse visualization from XY view
    - Different scales:
      * Accelerate with 2 wheels
      * Decelerate with 4 wheels
  + Purple region
    - Power limited in acceleration
  + Red dashed line:
    - Pure lateral cases
* Chart, funnel chart

  Description automatically generated
  + Red curve:
    - 2nd order polynomial because of downforce
  + Green arrow:
    - Loss of grip due to tire sensitivity
  + Blue curve:
    - Final performance curve
* Chart

  Description automatically generated
  + Green curve:
    - Braking performance
    - 4th degree polynomial with downforce and tire load sensitivity
  + Red region:
    - Acceleration is traction limited
  + Blue region:
    - Acceleration is power limited
    - Gear changes are visible
  + Orange curve:
    - Engine tractive force
* Final GGV map
  + Chart, surface chart

    Description automatically generated
    - Drag inclusion:
      * The whole map gets skewed towards negative
    - Red curve:
      * 2nd degree polynomial for the aerodynamic drag rolling resistance

Pure lateral case

* Diagram, schematic

  Description automatically generated
* Blue point:
  + Friction ellipse center
  + Wx moves center forwards and backwards
  + Fdrag moves center only rearwards
  + Wy moves center sideways
* Red point:
  + Pure lateral case without drag
* Green point:
  + Pure lateral case:
  + Throttle if:
  + Brakes if:

## Part IV: Track modeling in OpenTRACK

Track Generation Workflow:

* Excel/CSV input
  + Shape data
  + Logged data
* .mat output
  + Track output

Track Generation: 2 methods

* Shape data
  + Sheets
    - Info
    - Shape
    - Elevation
    - Banking
    - Grip factors
    - Sectors
* Logged data
  + Essential
    - Distance
    - Speed
    - Lateral acceleration
  + Optional
    - Yaw rate
    - Gryo (banking)
    - Altitude

Track settings

* Track information
  + Name
    - Set to the name of the track and the .mat filename
  + Country
    - Set to the country name
  + City
    - Set to the city name
  + Type
    - Permanent/Temporary/Street/Hill-climb etc.
  + Configuration
    - Set to “Closed” or “Open”
  + Direction
    - Set to “Forward” or “Backward”
  + Mirror
    - Set to “On” or “Off”

Track generation from shape data

* Type
  + Straight
  + Right turn
  + Left turn
* Length
  + Turn 🡪 Circular segment length
  + Segment sum = track length
* Radius
  + Assumed constant in section
  + Set to 0 for straight
* Adding more
  + Add values downward
  + Up to 10000 points
* Topology and grip
  + Point along track
  + Value
* Sectors
  + Starting point of sector
  + Sector number
  + 1st sector starts at 0
* Out of bounds
  + Values are linearly extrapolated
  + If distance > track length, then ignored
* Smoothing curvature
  + Chart

    Description automatically generated
  + Straights untouched
  + Corners via centers
* Smoothing long corners
  + A screenshot of a video game

    Description automatically generated
  + Long corner = long angle
  + Solution 🡪 points injection
* Injection length – κ

Track generation from logged data

* Essential channels 🡪 curvature
  + Essential channels
    - Speed (v)
    - Lateral acceleration (
  + Curvature
    - Radius
    - Turn direction
    - Curvature
* Optional channels
  + Yaw rate (
  + Gyro (banking) (
  + Altitude (Sensor or GPS)
* Alternative channels
  + Radius
  + Turn direction
  + Curvature
* Smoothing
  + Chart, line chart

    Description automatically generated

Track map generation

* Chart

  Description automatically generated with medium confidence
* Angles
* Next point

Track map shape adjuster **(IMPORTANT)**

* Graphical user interface, chart, line chart

  Description automatically generated
* Parameter

Track configuration

* Open
  + Chart, line chart

    Description automatically generated
  + No modifications
  + Enforces standing start in OpenLAP
* Closed
  + Chart, line chart

    Description automatically generated
  + Map gets connected
    - Connecting first ad last point
    - Start and end tangent
  + Topology gets modified linearly

Closing track: tangency

* Chart, line chart

  Description automatically generated 🡪Chart, line chart

  Description automatically generated
* Figure 8 detection
* Heading angle
* Section duration angle

Closing track

* Chart, line chart

  Description automatically generated 🡪 Chart, line chart

  Description automatically generated
* Closing loop

Closing map visualized

* Chart, line chart

  Description automatically generated

Map mirror, direction and rotation

* Mirror “On”
  + Chart, line chart

    Description automatically generated
* Direction “Backwards”
  + Chart, line chart

    Description automatically generated
* Rotation by
  + Diagram

    Description automatically generated with medium confidence

OpenTRACK code walkthrough (TODO)

1. Clearing memory
2. Track file selection
3. Mode selection
4. Settings
5. Reading file
6. Track model name
7. HUD
8. Pre-processing
9. Meshing
10. Map generation
11. Apexes
12. Map edit
13. Plotting results
14. Saving circuits

## Part V: Straight line simulation in OpenDRAG

Basic algorithm

* OpenDrag Simulates a vehicle in a straight line
* Lets assume that in time t, the vehicle is a point t with a forward velocity v and in a specific gear
* Let say the vehicle is trying to accelerate
  + Use drivetrain model to get the engine speed
  + Use the engine speed and engine torque curve to get torque output
  + Using drivetrain model again in opposite direction and tire model yields the wheel torque and acceleration tractive forces
  + Using the equation of motion yields the state of the vehicle in the next step
  + The process starts with zero speed and loops through until the maximum speed is reached, then it starts braking
  + When braking, the state of the vehicle is known. Using the tire model we can get the maximum braking forces. Then using the equations of motion, we can calculate the next state. This process repeats until the maximum speed is zero
* Graphical user interface

  Description automatically generated

OpenDrag Solver

* Time based, as opposed to OpenLAP’s distance solver
* Assume lateral acceleration is constant
* Accelerates vehicle from standing start, to max speed, then back to a stop
* Vehicle assumptions
  + No clutch
  + No load transfer.
    - Because point-mass
    - Could add bicycle model
* Road surface assumption
  + Grip factor = 1
  + Banking = 0
  + Inclination = constant
* Gear selection algorithm

1. First we check if the engine rpm is more than the shift point rpm
   1. If false, keep accelerating
   2. If true, either hit rev limiter, or shift one gear higher for more acceleration potential
2. Check if car is at highest gear
   1. If true, cannot gear up
   2. If false, initiate gear change
3. During gear change vehicle is decelerated a small amount due to drag forces, also have to wait for the specified delay to simulate the gear engagement time
4. When the gear is engaged, the car can continue acceleratingA picture containing diagram

   Description automatically generated

Drag limitation

* Chart

  Description automatically generated
* The vehicle has a gearing setting that allows it to go faster, but the tractive forces that it generates, is less than the drag force.
* The limit of the velocity when the time approaches infinity is equal to the maximum drag velocity
* At the same time that time approaches infinity, the longitudinal acceleration approaches zero
* Graphical user interface

  Description automatically generated
* **Problem:** as we decrease the time step to increase our accuracy, the solver needs more time to see that the velocity has not changed or that the velocity has reached zero.
* **Solution:** to solve the problem, we start accepting a small value of the longitudinal acceleration as our new zero value. This moves the red dotted line out by a small amount. This is done by checking if the longitudinal acceleration has become smaller than the sensitivity that we set.

OpenDRAG structure

* Start 🡪 Acceleration 🡪 Deceleration 🡪 End
  + Start
    - Get vehicle model
    - Define settings
    - Get const forces
    - Pre-allocate memory
    - Set boundary conditions
  + Acceleration
    - Previous state
    - Check rpm limiter
    - Check memory check drag limit
    - Speed-traps
    - Update forces
    - Update rpm
    - Gearshift
    - Get tps
    - Get long acc
    - Eqs of motion
    - New state
  + Deceleration
    - Previous state
    - Check if stopped
    - Check memory
    - Speed-traps
    - Update forces
    - Get gear
    - Get rpm
    - Get long acc
    - Get bps eqs of motion
    - New state
  + End

Code overview

* TODO

## Part VI: Lap time simulation

OpenLAP lap time simulation

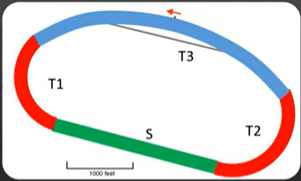
* Use OpenVEHICLE and OpenTRACK as input
* Outputs Matlab plots, .log file, and .csv export
* Diagram

  Description automatically generated

OpenLAP Workflow (**Read this if anything**)

* Read inputs and settings 🡪 Starts the simulation and calculates the maximum speed trace for the pure lateral position 🡪 Calculates the real apex positions 🡪 Sorts the apexes from slowest to fastest
  + Using the apexes in this way allows us to do as few calculations as possible without wasting resources solving for velocity values that would be omitted.
* Then accelerates the vehicle from all the apexes 🡪 decelerates the vehicle 🡪 post-processes the results 🡪 saves results and creates plots
* **Difference from other lapSims**
  + To find the apexes, OpenLAP finds the maximum speed trace for the pure lateral condition.

What is an apex?

* Michigan International Speedway example
* 
* Diagram

  Description automatically generated
* Straight colored in green, and two left turns colored in red, and one large radius turn colored in blue.
* If we look at the turning radius plot for this track, we can determine that the apexes would be at the local minima

Apex definition

* Definition in OpenLAP
  + Has just stopped decelerating
  + Is at a pure lateral condition
  + Start accelerating right after
* May appear when
  + Curvature local maxima
  + Grip factor local minima
  + Slope or banking change
* How to find
  + Local minima of max speed
* Chart, line chart

  Description automatically generated
  + Blue: maximum velocity trace
  + Black circle: local minima
  + Orange: final solution
  + Shows vehicles inability to use all its lateral performance everywhere
  + Driver tries to move arbitrary green speed curve upwards
    - Think of the speed trace as a rope that is allowed to touch and wrap around bumps created by the blue curves, but is not allowed to move them
    - This makes it hang in air between the bumps
    - This shows us the vehicle’s combined performance
  + Engineers try to move the blue curves upwards and make the rope more flexible
    - this increases the area underneath it, i.e. the vehicles performance

Maximum speed trace

* OpenLAP uses its vehicle model lat function
* Function
  + Vehicle\_model\_lat()
  + Get pure lateral condition
  + Solves each point independently
* Straights
* Corners
  + Diagram

    Description automatically generated
  + Get red point
  + Correct for green point

Maximum Speed Trace MAX LATERAL

* Driving equations
  + Algebraically combine
* Final equation
  + Using ) we get the initial max lateral guess

Maximum Speed trace PURE LATERAL

* Processes this loop:

1. Obtain
2. Calculate
3. Calculate
4. Calculate
5. Calculate via friction ellipse
6. Check
   1. If pass, then
   2. If fail, move to 8.
7. New:
   1. Loop back to 1.

Maximum speed trace

* The full equation