Developing Optimal Power Split Controls in Hybrid-Electric Vehicles through Dynamic Programming Optimization

Electronics Project Internship Report

Asgard Kaleb Marroquin

Technische Universität Dresden, Germany

Summer 2016

Acknowledgements

I would like to especially thank Stephan Uebel, Robert Kloss, Conny Tempelhahn, and Prof. Dr.-Ing. Bernard Baeker for the opportunity to work on such a fascinating subject this summer. Their mentoring, supervision, and advice was instrumental in understanding and developing optimal control theory this summer. I also thank all those at the Innovationzentrum Fahrzeugmechatronik, whose openness and inclusiveness helped me integrate into the mechatronics team professionally, academically, and culturally.

Preface

My internship this summer took place in Dresden, a small city on the east side of Germany, where industry and STEM-related technical training have taken strong root. In particular, I interned with the Innovationzentrum Fahrzeugmechatronik, or the Department of Vehicle Mechatronics, at the Technische Universität Dresden. The department works directly with the transportation agency in Dresden – the Dresdener Verkehrsbetriebe (DVB) – in reducing fuel consumption and heat losses in the city's ever-growing hybrid-electric bus fleet. The group has also collaborated with private companies including BMW, Daimler, Honda, Volkswagen, Continental, Evobus, and MAN. The department investigates various vehicular aspects, such as power optimization strategies for powertrain management, thermal losses from passenger exchange and smaller engine sizes, bus log data analysis, battery technologies, real-time component diagnostics, and conductor assistant systems for achieving more efficient driving cycles. This summer I worked on a project in collaboration with Continental evaluating online energy management systems by calculating the benchmark global optimal fuel consumption and power split for a hybrid electric vehicle through dynamic programming.

Introduction

WRITE INTRODUCTION TO HEVS – necessary?

Various control methods have been investigated for optimizing energy management systems, or EMSs, for HEV powertrains. Two main control strategies exist for developing optimal HEV powersplit strategies: numerical methods for known *a priori* speed trajectories demand profiles *(*noncausal) and

analytical methods calculating without such knowledge (causal) (Jager, van Keulen, and Kessels et al 2013; Elbert 2013). Noncausal methods, requiring perfect future driving condition predictions, are able to compute across the entire given driving profile and find globally optimal solutions for EMSs (Tempelhahn 2016). Were noncausal optimization methods be used in realtime, stochastic inputs deviating from the defined driving cycle would influence calculations; as a result, such methods are typically performed offline and are used as benchmarks for comparing and designing efficient online causal EMSs (Tempelhahn 2016). Causal methods performing online, on the other hand, can only optimize locally portions of the driving cycle problem and thus return suboptimal strategies (Tempelhahn 2016).

Dynamic programming (DP) is typically used for calculating globally optimal EMS solutions due to its flexibility and ability to ensure global optimality. DP can utilize both mixed-integer and continuous state variables with nonlinear and nonconvex models, conditions which other methods cannot numerically make function (Elbert 2013; Murgovski et al 2013). However, dynamic programming can only practically optimize an EMS using a limited number of state variables because computational time increases exponentially as more state permutations for a given driving profile are analyzed – i.e. the "curse of dimensionality" (R. Bellman, *Dynamic Programming*, Princeton, NJ. USA: Princeton Univ. Press, Jun. 1957). DP also requires predetermined travel profile data and is thus noncausal by nature. Nevertheless, due to the repetitive, predictable nature of city bus routes, speed profiles for hybrid-electric transit can be approximated based on past logged data records. Standard driving trajectories, such as the New European Driving Cycle, attempt to mimic urban driving speed profiles and are typically used for determining typical urban emissions (INSERT REFERENCE); these trajectories can also be used as a baseline for testing HEV EMSs.

Alternative causal optimization methods such as convex optimization and methods utilizing Pontryagin's maximum principle (PMP) are also researched in order to implement online control systems in HEVs. Although far faster than DP, these methods naturally have their limitations. For example, convex optimization requires a linear convex model utilizing continuous state variables in order to function effectively. Should a problem be originally nonconvex, it must be manually made convex, possibly affecting optimality results due to model simplification (Tempelhahn, Conny 2016). And although mixed-integer boolean state variables – engine on-off control, for example – can be theoretically relaxed to become valid as continuous between zero and one (Murgovski, Nikolce et al 2013), it was agreed upon by the research group that smoothing out strict boolean controls may yield questionable results at best and the proceudure may not yet be generally accepted as sound. PMP-based strategies are also limited to processing only continuous, unrestricted state variables for short

time/space horizons, but can also find near-globally optimal results for such variables at an exponentially faster rate than dynamic programming (Tempelhahn, Conny 2016).

Due to the strengths and weaknesses of various different optimization control methods, they have been combined in order to compliment with each other's characteristics. Faster procedures such as convex optimization and PMP can be integrated into DP algorithms and optimize the continuous variables in the model. This frees up the number of states DP must calculate and dedicates the algorithm to optimize just mixed-integer states, considerably reducing computation time as a result (Uebel, Stefan et al 2016, Murgovski, Nikolce et al 2013). Should an accurate DP-PMP combined method become quick enough to be implementable online, near optimal EMSs would improve on-road HEV fuel efficiencies. However, as concluded by Uebel et al, this method is not yet fast enough to be implemented online and must be further investigated.

Two main projects were defined for this summer internship. In an attempt to speed up DP-PMP calculations for online testing, my first task involved further speeding up and converting the MATLAB/MEX DP-PMP algorithm developed by my supervisor into C/MEX so that it may be tested in a Autobox, a rapid control prototyping device. This algorithm uses the continuous variables vehicle kinetic energy, battery electric energy, and time, as well as discrete gears, as state variables to minimize fuel use for an articulated bus model making control decisions at a spatial resolution of every ten meters. The details of this control methods can be found in (Uebel, Stefan et al 2016). Although the port was successful, I found no noticeable improvement in calculation time when comparing the C/MEX port to the original MATLAB/MEX script to make the method viable for online testing. This is most likely as a result of the large amount of optimization that has already gone into running MEX functions so that their computation speed is essentially the same as an equivalent C script (not to mention the extraordinary convenience of also avoiding having to manually port code). Nevertheless, the project aided in learning the fundamentals of dynamic programming and modern controls through “hands-on” experience with DP, and proved invaluable for completing the second project described in this report.

More relevant to working with controls, my second task involved developing a new DP algorithm which would find the dynamically optimal power split control of a Ford Focus-like HEV model based on battery energy levels, discrete gears, and boolean engine on-off states. The goal of developing this dynamic programming method was to create a globally optimal benchmark with which future online strategies could be compared with. The temporally resolved offline model, requiring a predefined driving profile, is based on the New European Driving Cycle. By developing a DP algorithm based on modern control theory which models and dynamically optimizes battery use in a

hybrid-electric bus at the macroscopic level, this internship has exposed me to the field of controls, a fundamental field in electrical engineering.

In order to describe the project in detail, the vehicle model utilized this summer is first described. The state variables – battery energy, gear, and engine state –and the battery energy state dynamics alongside gear and engine control signals defining changes in the state variables are also defined. The vehicle model and state variables are combined in the description of the DP model minimizing the given cost function – that is, fuel use. Finally, results from the algorithm showing optimal trajectories for various possible final ending battery energy states are shown and discussed.

Problem Description

New European Driving Cycle

The New European Driving Cycle (NEDC), a speed profile mimicking a typical commute in a European urban environment, is used as a vehicular travel profile for the DP algorithm (Figure 1). The driving cycle consists of four Urban Driving Cycle (UDC) periods of driving simulating sporadic low vehicle speed and engine load experienced in cities, followed by an Extra Urban Driving Cycle (EUDC) simulating faster, more aggressive travel (United Nations Environment Programme).

State Variables

The HEV is being optimized for minimal fuel use based on three state variables: battery buffer state of charge (SOC) energy level E_B, gear selection g, and engine on-off decision e (-2).

$$\boldsymbol{x}=\langle E_B, g, e\rangle \quad (-2)$$

Control signals for the discrete gear and engine states are also defined as an input vector (-1).

$$\boldsymbol{u}=\langle u_g, u_e\rangle \quad (-1)$$

The battery energy level E_B, a continuous variable in reality, must be discretized so that the state variable can exist in the DP state space. The finer the resolution, less overshoot will occur and the algorithm would yield more accurate results. However, more finely resolved changes in E_B require exponentially more computing time, due to the increased number of possible E_B states and thus must check additional state variable permutations. Therefore, a battery energy level resolution ΔE_B of 2500 J is chosen in order to attempt to balance these two factors.

Vehicle Model

In order to optimize the power split control in the HEV, a vehicle model able to satisfy the wheel torque demand satisfying the NEDC must be defined. Vehicle model parameters, (eg vehicle mass, battery power and energy limits, internal combustion engine (ICE) and electric motor (EM)

speed and torque bounds, etc), in this model are based from a hypothetical Ford Focus HEV.

Defining the power demand split for an HEV will depend on the vehicle's drivetrain architecture. The model works with a parallel drivetrain, in which the ICE and the EM are completely decoupled. This permits both machines to independently contribute to the requested crankshaft power demand $\boldsymbol{P_{CRS}}$ (Jager, van Keulen, and Kessels 2013).As engine on-off control e is a state being optimized for in this EMS, the ICE can either contribute power $\boldsymbol{P_{ICE}}$ to (e=1) or be disengaged from (e=0) the crankshaft, leaving the EM to supply enough power $\boldsymbol{P_{EM}}$ to completely satisfy $\boldsymbol{P_{CRS}}$ for the given time index decision moment (0). Should a given P_{ICE} exceed the P_{CRS} demand, the excess power can go back into recharging the battery buffer (Figure 2).

$$\boldsymbol{P_{ICE}}(\boldsymbol{x}) = \begin{cases} \boldsymbol{P_{CRS}}(g) - \boldsymbol{P_{EM}}(E_B) \; if \; e=1 \\ 0 \; if \; e=0 \end{cases} \quad (0)$$

To calculate the requested power demand, the crankshaft speed and the sum of forces at the wheel must first be determined. The vehicle's velocity vector $\boldsymbol{v}$, wheel constant rolling radius r, and gear ratio $\gamma(g)$ define the crankshaft's rotational velocity $\boldsymbol{\omega_{CRS}}(g)$ (1), where $\boldsymbol{v}$ is the NEDC speed profile discretized every second.

$$\omega_{CRS}(g) = \frac{\boldsymbol{v}}{r}\gamma(g) \quad (1)$$

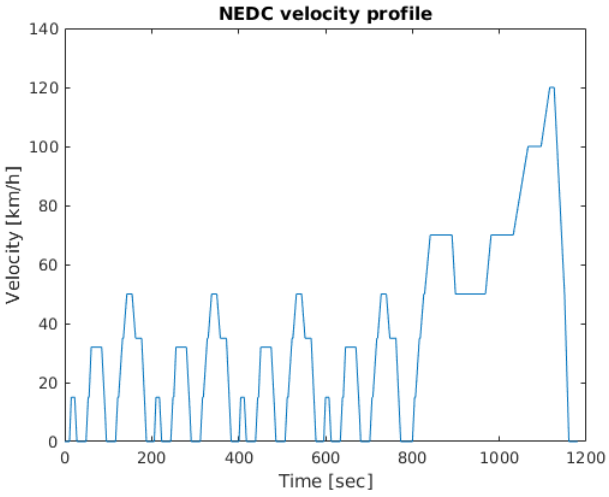

Figure 1: The New European Driving Cycle (NEDC), the velocity profile used in this project's DP optimization. **REFERENCE HERE**

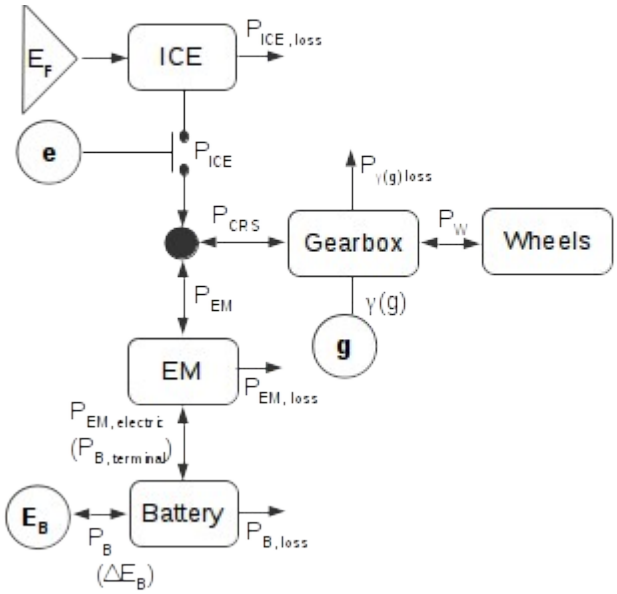

Figure 2: Model parallel HEV drivetrain used. The three state variables (shown as circles) affect how the ICE and EM satisfy the power demanded by the wheels. Dynamic programming analyses state variables possible values and, by controlling their state dynamics at discrete points of time along the NEDC, cumulative fuel use (shown as a triangle) is optimized. Figure based on (Jager, van Keulen, and Kessels 2013)

The forces at the wheel can be described though use of Newton's Second Law (2), INCLUDE TIME

$$m\frac{d\boldsymbol{v}}{dt}+\boldsymbol{c_\alpha}+\boldsymbol{c_\beta}+c_a\boldsymbol{v}^2+F_B=(\boldsymbol{F_{WHEEL}})\gamma(g) \quad (2)$$

where m is the vehicle mass, $d\boldsymbol{v}/dt$ is vehicular acceleration, $\boldsymbol{c_\alpha}= m*g*\sin(\boldsymbol{\Theta_{gradient}})$ is the vehicle's gradient force, $\boldsymbol{c_\beta}= \mu_{drag}*m*g*\cos(\Theta_{gradient})$ is the vehicle drag force, $\boldsymbol{\Theta_{gradient}}$ is the speed trajectory's slope profile (defined here flat). F_B are braking forces, and $\boldsymbol{F_{WHEEL}}$ is the amount of force the wheel must deliver.

The wheel torque demand $\boldsymbol{T_{WHEEL}} = \boldsymbol{F_{WHEEL}}*r$ for the NEDC is found and subsequently the crankshaft torque $\boldsymbol{T_{CRS}}$ (3) and power $\boldsymbol{P_{CRS}}$ (3a) are derived. The powertrain split satisfying $\boldsymbol{P_{CRS}}$ between the internal combustion engine's torque $\boldsymbol{P_{ICE}}$ and electric motor's torque $\boldsymbol{P_{EM}}$ is what is optimized for this HEV's EMS during DP calculations.

$$\boldsymbol{T_{CRS}}(g)=\begin{cases}\dfrac{\boldsymbol{T_{WHEEL}}}{\gamma(g)}*\eta_g \; for\; \boldsymbol{T_{CRS}}(\boldsymbol{T_{WHEEL}}<0)\\ \dfrac{\boldsymbol{T_{WHEEL}}}{\gamma(g)}/\eta_g \; for\; \boldsymbol{T_{CRS}}(\boldsymbol{T_{WHEEL}}\geq 0)\end{cases} \quad (3)$$

$$\boldsymbol{P_{CRS}}(g)=\boldsymbol{T_{CRS}}(g)*\omega(g) \quad (3a)$$

Battery Model

Describing the effects of E_B's state dynamics (4) on the crankshaft power demand requires a battery model illustrating the intermediate processes (Figure 3).

$$E'_B=f_t(E_B)=P_B*\Delta t \quad (4)$$

Rearranging (4), using a select possible present and predecessor E_B value yields a battery power $P_B(t)$ for the given time index (5).

$$P_B(t)=\frac{E_B(t)-E_B(t-1)}{\Delta t} \quad (5)$$

In this model, dissipation conversion losses $P_{EM,\,loss}$, input auxiliary power P_{AUX}, and internal battery losses $P_{B,loss}$ are taken into consideration when translating a change in the state E_B into P_{EM} (6).

$$P_{EM}(t)-P_{EM,loss}=P_{EM,electric}(t)=P_B(t)-P_{B,loss}-P_{AUX} \quad (6)$$

Electric battery power losses can be easily derived from Ohm's Law. The amount of electric power transferred to the EM is simply the difference between the battery's terminal power (7) and the battery's internal losses (8).

$$P_B=V(E_B)I \quad (7)$$

$$P_{B,loss} = I^2 R(E_B) \quad (8)$$

If a certain E_B level is being evaluated, as is the case when looping through various E_B values in DP, the battery current I can be easily calculated from (7), allowing for a straightforward $P_{B,loss}$ calculation in (8). Otherwise, if a sample $P_{EM,electric}$ is given instead (for example, during preprocessing when setting bounds on reachable E_B states), equation (6) must be rearranged so that the battery current I can be calculated quadratically and substituted back into (8), as shown in (9).

$$P_{B,loss} = \frac{V(E_B) - \sqrt{(V^2(E_B) - 4RP_{EM,electric})}}{4R} \quad (9)$$

The battery's open circuit voltage V is nonlinearly dependent on E_B. V is derived from a model input lookup table. **INCLUDE GRAPH OF OCV WRT SOC?** The internal resistance R will have a different sign (and possibly a different value) if either the input P_B or $P_{EM,\ electric}$ (depending on how I was calculated, during DP or preprocessing, respectively) is charging or discharging the battery (10).

$$R = \begin{cases} R_{CHARGE} \ for (P_B > 0 \vee P_{EM,electric} > 0) \\ R_{DISCHARGE} \ for (P_B \leq 0 \vee P_{EM,electric} \leq 0) \end{cases} \quad (10)$$

State Variable Bounds

Bounds must also be placed on the state variables due to physical limitations. Because the entire

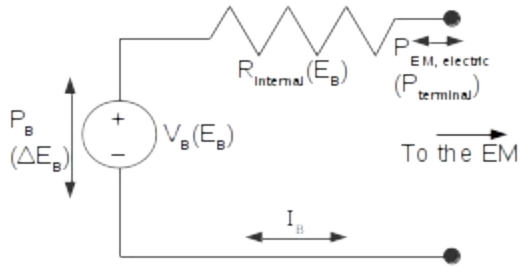

Figure 3: Battery model circuit.

speed demand profile is known ahead of time, various bounds for E_B can be calculated beforehand. Two bounds for E_B's state dynamics ($P_{B,MIN}$ and $P_{B,MAX}$) are preprocessed: power limitations based on the HEV model's EM speed limits $\boldsymbol{P_{EM,\ MIN}(\omega_{CRS},\ t)}$ and $\mathbf{P_{EM,\ MAX}}(\boldsymbol{\omega_{CRS},\ t})$, and power limitations based on the model battery's energy charge/discharge limits P_{BAT}. Gear changes u_g are also bounded so that the gear can only transition to adjacent states from one time interval to the next.

It should be noted that at the time of this report that I am analyzing whether $P_{EM,\ MIN}(\cdot)$ and $P_{EM,\ MAX}(\cdot)$ referred to mechanical or electrical power limitations. I originally treated it as an electrical power limitation based on rotational velocity limitations of the EM. Should however these parameters refer to physical limits instead, an additional power conversion term during preprocessing must be added to account for losses. I plan to address this issue with my mentor in the very near future in order to ensure

optimal results. This issue occurs only when working with an input $P_{EM,\,electric}$ when preprocessing, and not when running through E_B states when running DP (that is, when working through equation (6) left to right instead of right to left). Physical EM torque T_{EM} can be interpolated from a lookup table based on ω_{CRS} and P_B (derived from g and E_B states, respectively); because the the nonlinear conversion loss is included T_{EM} interpolation, the physical EM power $P_{EM}(E_B, g) = T_{EM}(E_B)*\omega(g)$ can be derived when E_B is provided.

This preprocessing drastically reducing computational time while running DP, as various physically unreachable states were immediately ignored during calculations. In this DP algorithm, calculation time dropped drastically after implementing preprocessed bounds from 24 hours to just over fifteen minutes. Defining these bounds beforehand for all possible crankshaft speeds and torques through time removes the rote calculation otherwise required in DP when bounds would have to be calculated for each time interval and nested previous and current state permutation loops.

Objective Function

The objective of this EMS is to find the path burning the least amount of fuel for the NEDC trajectory. Therefore the objective function (11) is the integral of the ICE's power along the path with respect to time, along with any penalties defined for switching gear (β_g) or engine (β_e) states.

$$minimize\ E_{FUEL} = \int_{t_i}^{t_f} ((P_{ICE}(\boldsymbol{x}, \boldsymbol{u}) + P_{ICE,loss}) + |u_g|\beta_g + |u_e|\beta_e)\, dt = \int_{t_i}^{t_f} \Lambda(\boldsymbol{x}, \boldsymbol{u})\, dt \quad (11)$$

Because this function must analyze both the change in E_{FUEL} as well as the cumulative sum of all previously selected E_{FUEL} values at every time interval t, at each change in E_{FUEL} for the current time index t, all previous reachable states $\boldsymbol{x}(t-1)$ (with their respective control signals $\boldsymbol{u}(t-1)$ and state dynamic $f(E_B(t-1))$ must be calculated for every given state $\boldsymbol{x}(t)$. Thus, in order to minimize the objective, the recursive feedback effects of selected predecessor states $\boldsymbol{x}$ for every optimally calculated E_{FUEL} must be taken into consideration; otherwise, the algorithm is greedy and will not ensure a globally optimal result (D. P. Bertsekas, Dynamic programming and optimal control).

Dynamic Programming Implementation

Now that the state variables, battery dynamics, gear and engine control signals, vehicle and battery models, and a objective cost function to minimize for have been defined, dynamic programming can be used to run through each realistic state variable permutation through the use of nested for-loops. DP is based on Bellman's Principle of Optimality, where, for a given problem which can be broken up into smaller subproblems (a characteristic the problem must have for DP to work), the globally

optimum path in the problem be comprised of smaller optimal paths along its intermediate steps, as long as the cost function is additive over time (R. Bellman, *Dynamic Programming*, Princeton, NJ. USA: Princeton Univ. Press, Jun. 1957).

Therefore, given initial and final state conditions, DP can be ran forwards or backwards in time and calculate the optimal trajectory either way for a given set of initial or ending state variable **x** conditions, respectively (12e),. In this model, DP runs forward with intermediate steps Δt spaced one second apart in time (12a). The number of possible changes in the continuous state variables E_B for every time index t is limited by the discretized state dynamic bounds (12b) defined during preprocessing, The discrete states g and e are also subject to the control signals $\boldsymbol{u}$ triggering at every decision moment t (12c, 12d), where $\boldsymbol{u}$ is also limited due to physical restrictions (12g 12h).

$$minimize\ \Lambda_{t,f}(\boldsymbol{x}) + \sum_{k=t_i}^{t_f-1} \Lambda_k(\boldsymbol{x},\boldsymbol{u})\,\Delta t \quad (12a)$$

$$f(E_B) \in [P_{B,MIN}, P_{B,MIN} + P_{B,STEP}, \ldots P_{B,MAX}] \quad (12b)$$

$$g(t) = g(t-1) + u_g \quad (12c)$$

$$e(t) = u_e \quad (12d)$$

$$\boldsymbol{x}(t_i) = \boldsymbol{x_i}, \boldsymbol{x}(t_f) = \boldsymbol{x_f} \quad (12e)$$

$$\boldsymbol{x} \in [\boldsymbol{x_{min}}, \boldsymbol{x_{max}}] \quad (12f)$$

$$u_g = \{-1, 0, 1\} \quad (12g)$$

$$u_e = \{0, 1\} \quad (12h)$$

The dynamic programming algorithm functions by using nested for-loop to loop through every possible current state variable combination ***x(t)*** lying within bounds (12f). At each state **x**(t), all predecessor state variable combinations ***x(t-1)*** satisfying the criteria set in (12b), (12g), (12h) are looped through, and each respective predecessor state variable's output fuel energy use E_{FUEL} is calculated and stored in a temporary three-dimensional tensor (one dimension per state variable). The optimal **x**(t) which most minimizes $\Lambda(t)$ is then added into an ongoing cumulative cost tensor in a process called memoization. Memoization, or saving the optimal solutions of subproblems, is a part of what makes DP so efficient, as it eliminates the need to recursively calculate the values of all previous subproblems to reach the current problem by instead loading the additive cumulative sum of said previous subproblem costs into a tensor.(INSERT REFERENCE).

Running this process for every time time interval t from $[t_{i+1} : t_f]$ creates a map of nodes and edges, where each node represents a state variable permutation at a specified point in time and weights connecting nodes across adjacent time intervals representing the optimal E_{FUEL} value for each

subproblem. Finally, after generating this map of all possible E_{FUEL} weights, running through this map selecting the saved optimal **x** values (and therefore the optimal $f(E_B)$ and $\boldsymbol{u}$) though time for specified end conditions (12e) yields t trajectory's optimal minimal cumulative fuel use $_{EF}$, effectively applying Bellman's Principle of Optimality.

Results

Conclusion

Two main projects

References

Elbert (2013)

United Nations Environment Programme