Principles of Mechatronic System Analysis and Design

or

“Striking a Balance is Making Everybody Equally Unhappy”

Paul Pounds

11 March 2014
University of Queensland
But first…

Some house keeping
Timezones…

1. Brisbane time is (approximately): 12:00 pm
2. Dublin time is (approximately): 2:00 am
3. Guinness in Dublin is surprisingly affordable
Zzz....

Please refrain from:
- Loud, sudden noises
- Bright and blinding lights
- Intellectually challenging questions
House keeping

• Still taking team exclusion requests
  – Requests will close on Friday

• No pracs this week, either
  – Just focus on design analysis
  – You have been thinking about that, right??
Calendar at a glance

<table>
<thead>
<tr>
<th>Week</th>
<th>Dates</th>
<th>Lecture</th>
<th>Reviews</th>
<th>Demos</th>
<th>Assessment submissions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2/3 – 6/3</td>
<td>Introduction</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>9/3 – 13/3</td>
<td>Principles of Mechatronic Systems design</td>
<td></td>
<td>Problem analysis</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>16/3 – 20/3</td>
<td>Professional Engineering Topics</td>
<td></td>
<td>Analysis peer review</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>23/3 – 27/3</td>
<td>Your soldering is (probably) terrible</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>30/3 – 3/4</td>
<td>???</td>
<td>Progress review 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Break</td>
<td>6/4 – 10/4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>13/4 – 17/4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>20/4 – 24/4</td>
<td></td>
<td>Progress seminar</td>
<td>25% demo</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>27/4 – 2/5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>4/5 – 8/5</td>
<td></td>
<td></td>
<td>50% demo</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>11/5 – 15/5</td>
<td></td>
<td>Progress review</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>18/5 – 22/5</td>
<td></td>
<td></td>
<td>75% demo</td>
<td>Preliminary report</td>
</tr>
<tr>
<td>12</td>
<td>25/5 – 29/5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>1/6 – 5/6</td>
<td>Closing lecture</td>
<td></td>
<td>Final testing</td>
<td>Final report and reflection</td>
</tr>
</tbody>
</table>

You are here

Switch to Q and A sessions

Next deliverable
Begin the terror

- Problem Analysis is due *Friday*
- First progress review is three weeks after that

DON’T PANIC

But do get started!
Managing your stress levels

- Some students expect this class will be stressful – this does not have to be the case!
Some points for perspective

• Problem analysis is only two pages
  – A lot of thinking, but not a lot of “work”
  – I will talk more about it at the end of the class

• Progress reviews are to get you moving early, not to cause you stress
  – Make a reasonable effort and produce some tangible output and you’ll be fine (no ‘tales’)

FAQ Roundup

• **Can we make a VTOL?**
  – Nothing in the spec says you can’t. But seriously, that’s going to be hardcore; I’ll be impressed if you succeed! Also, please note the requirements about propulsion systems requiring integrated safety ducts.

• **So, just to be clear.... no practicals or contacts until next week?**
  – Yep. None last week or this week. And also, contact sessions aren’t really a ‘thing’ per se
Lab orientation sessions

• *Next week, Wednesday 18th of March, 3 pm in Hawken c404*
  – I always said it was on the 18th, and anyone who says otherwise is an **enemy of the state**

• It is important you attend:
  – Room safety induction
  – Hand out tool boxes to teams
  – Great time to meet your other group members
Back to that design thing…

Mechatronic Systems Design

Lolwut?
What is design, anyway?

• Design:
  – n. A goal or intention
  – v. The process or creating a plan

• In engineering contexts, design is both the process and the end product of formulating a technological solution to a problem
  – Engineering design is the application of scientific knowledge to satisfy a goals
Things that are designed

- Devices/structures
- Materials/chemicals/substances
- Processes/formulae
- Documentation/procedures/policies
- Specifications/guidelines/standards
- etc.

The common thread:

“Things that make stuff work”
Design is purposeful

• All design has some end goal

  eg.
  – Providing power to 100,000 homes
  – Moving 1 Megahuman across a city twice a day
  – Putting a catarrhine into orbit
Design is constrained

• All design is constrained
  – With no constraints – no limits – anything is achievable without need for planning
  – Design is needed when failure is possible
    contra: no need to ‘design’ trivial things

• Common constraints:
  – Time          – Materials          – Machine access
  – Budget        – Energy            – Technology
  – Labour        – Logistics          – Political capital
A way of thinking about design

Design is the dual of critique:

**Analysis**
- Specification
- Deconstruction
- Parameterisation

**Synthesis**
- Implementation
- Constitution
- Optimisation

These are tools and philosophies for thinking about design, not a cookbook or an excuse not to use your brain.
Laying it out

How do we know what it is we have to design?
Specifications

• The precise statement of exactly what the system will do
  – Often worked out collaboratively with the engineering team

• Precision is key
  – Reduce uncertainty as much as possible
  – Avoid “feature creep”
  – Clients often don’t know what they want
    (and sometimes change their minds halfway!)
Design brief

• Design briefs communicate the specification of an engineering problem to the engineer

• Describes what must be done
  – Provides precise requirements and constraints
  – Specify metrics to assess success

• Preliminary analysis of the problem
  – Theoretical design implications
  – Possible solutions, their risks, benefits, issues
Specifications are important

• All of your design effort is geared to meeting the specification
  – Avoids putting effort into unnecessary areas
  – Clear, complete specs’ lead to better designs
  – Doubles as a performance claim to customers

• In legal disputes, meeting the specification can be a critical defence against breach of contract
And now a word from our sponsor…

A brief detour into systems thinking
So what about systems?

- A system is a set of interrelated elements that interact as a whole
  - eg. transport networks, computers, duct tape, people, vehicles, weather

Systems Engineer maxim: “Everything is a system”
Systems engineering

• Engineering the whole, rather than the parts
  – Structured way of handling complexity
  – Defines the interfaces between major components of the system
  – Abstracts performance and robustness from individual parts towards the gestalt

• Design uses systems to modify the state and behaviour of other systems
The systemic approach

- Systems decompose into networks of subsystems with touching boundaries
  - Information crosses the system interface
System interfaces

• All systems have interfaces, designed or otherwise – eg. a box resting on a table

• Good design makes interfaces explicit
  – Software library APIs
  – ATX mounting holes and dimensions
  – IAC 240V/10A plugs and sockets
  – Road rules, air traffic control

• Interface designs are mutual agreements
  “If you function like this, I will work with you”
Systems get complex *fast*

- Large systems like space shuttles and skyscrapers and Olympics opening ceremonies are fiendishly complex
  - Even small systems (e.g. cellphones) can be overwhelming in their entirety

How can we possibly understand them?
Deconstruction

• Specifications provide an end-point for the design process
  – Often work backwards to find a starting point

• Reductionism: break complex things down into understandable pieces
  – Find the essential parts of a system
  – Distil the problem into the core challenges
Deconstruction

• Deconstruction is like systems disassembly:
  Pull things to pieces to find out how they work together

• Many ways of doing this
  – Functional decomposition
  – Process flow
  – Causal dependency trees
  – And many more…
Functional* decomposition

• Hierarchical arrangement of processes
  – Interconnected network of “stuff that is done”
  – Not necessarily in order of operation

*Functional as in “Thing that make things work”, not as in void main(void);
Causal system dependency

- Systems can be thought of as a cascaded series of enabling functions

![Diagram showing causal relationship with terms Prerequisite and Dependent]
Causal system dependency

• For example:
Causal system dependency

• For example:
Causal system dependency

• For example:

Increasing level of detail
Causal system dependency

- For example:
Causal system dependency

• For example:

- Engines
- Energetics
- Cryogenics
- Trajectory
- Monkey in space

Increasing level of detail
Causal system dependency

- For example:

![Diagram showing causal system dependency]

Increasing level of detail
Causal system dependency

- For example:

Increasing level of detail
Causal system dependency

• For example:
Causal system dependency

• For example:

Increasing level of detail
Causal system dependency

• For example:

- Power bus
  - Launch stack
  - Cryogenics
- Engines
- Sensors
- Flight control
- Energetics
- Trajectory
- CO₂ scrubbers
- Heaters
- Life support system
- O₂ supply
- Monkey in space
Causal system dependency

- For example:

Power bus

Launch stack

Cryogenics

Engines

Sensors

Flight control

Energetics

CO$_2$ scrubbers

Life support system

Heaters

O$_2$ supply

Monkey in space

Trajectory
Aid to understanding failure

• We can also use causal system decomposition to understand failure

• Faults in the system propagate through directed graphs
  – Find the consequences of a failure
  – Work upstream to find the causes of a failure
  – Verify the “causal chain” to prove the system
Aid to understanding failure

• Consider a system defect:

- Engine
- Sensors
- Flight control
- Energetics
- Trajectory
- CO₂ scrubbers
- Heaters
- Life support system
- O₂ supply
- Monkey in space
- Power bus
- Cryogenics
- Launch stack
Aid to understanding failure

- Consider a system defect:

- Engines
- Sensors
- Flight control
- Energetics
- Trajectory
- CO₂ scrubbers
- Life support system
- Heaters
- Monkey in space
- Power bus
- Launch stack
- Cryogenics
- O₂ supply
Aid to understanding failure

- Consider a system defect:

![Diagram showing system components and failure path]

- Engines
- Sensors
- Flight control
- Energetics
- Trajectory
- CO₂ scrubbers
- Power bus
- Cryogenics
- Life support system
- Heaters
- O₂ supply
- Monkey in space
- Launch stack
Aid to understanding failure

• Consider a system defect:

 ![Diagram of a system defect]

- Engines
- Sensors
- Flight control
- Trajectory
- CO₂ scrubbers
- Cryogenics
- Power bus
- Heaters
- Life support system
- O₂ supply
- Monkey in space
- Launch stack
Aid to understanding failure

• Consider a system defect:

- Engines
- Sensors
- Flight control
- Energetics
- Trajectory
- Power bus
- Cryogenics
- CO₂ scrubbers
- Heaters
- Life support system
- Monkey in space
- O₂ supply

Launch stack
Aid to understanding failure

- Consider a system defect:

  ![Flowchart Diagram](Image)

  - Power bus
  - Cryogenics
  - Launch stack
  - Engines
  - Sensors
  - Flight control
  - Energetics
  - Trajectory
  - CO\(_2\) scrubbers
  - Life support system
  - Heaters
  - O\(_2\) supply

  Monkey will not go to space today
Deconstruction informs design

- System decomposition tells us what functions are required by the design
- Successful design satisfies all prerequisites
- Robust designs provide redundant pathways
  - Identifies “weak links”
Constitutive design

• ‘Constitution’ turns functions into features
  – Tells you how the big picture fits together
  – The “broad strokes” of defining an approach
  – Functional requirements are a guide at best

• Designers have the most flexibility during design constitution – also the highest risk!
  – Bad structural decisions effect everything else
  – Constitution determines ~90% of system costs
Putting the pieces together

- Eg. Monkey capsule must provide oxygen, remove CO$_2$, and regulate temperature.
  - How big do the oxygen tanks need to be?
  - Can we use standard gas tanks or do they have to be custom-built?
  - How much separation is needed between the O$_2$ and the heater elements?
  - How do you get the monkey in there, anyway?
A jigsaw puzzle

- Engineering is just like solving a jigsaw puzzle with many pieces, except each piece costs $1000 and can be one of a dozen shapes or completely custom-made, and if you don’t solve the puzzle right, people die.

- In real life, engineering is often a process of try-and-see iteration
  - Sometimes, there is no “right” way.
Parametric design

• Even when you *do* have a clear high-level structural concept of your solution, there will usually still be many unconstrained variables

• The key dimensions/values/settings that describe a design are called the “design parameters”
Parametric design

- Parameters can be thought of as a series of twist knobs that adjust the design
Design space

- We can think about a particular configuration of parameters as a point in “design space”
Design space

• Combinations of design parameters map to some realisable performance of the system
The fundamental problem

• Knowing how to set those knobs is difficult
  – Complex interactions between parameters
  – Competing design goals
  – Constraints on parameter space

• You will rarely satisfy all of your goals
  – You will NEVER meet all your ambitions
Design constraints

• University engineering problems are typically tightly constrained – they have only one “right” answer

• In the real world, engineering problems are either under-constrained (many solutions) or over-constrained (no solutions)
Design constraints

“Design Space” (Parameter space)  
Impossible configurations  
$P_3$  
$P_2$  
$P_1$  
$P_n$

“Performance Space”  
Outside of specification  
$X \text{ m/s}$  
$Y \text{ kg}$  
$Z$

“Design Space” (Parameter space)  
Impossible configurations  
$P_3$  
$P_2$  
$P_1$  
$P_n$
Design metrics

- We can make cost and value functions to encode how “good” a candidate design is
  - eg. Aim to maximise propellant bang for buck, given parameters 1 to $n$, we might use:

  \[
  \text{Value} = \frac{\text{explosive force}}{\text{unit cost}}
  \]

  \[
  V = \frac{X(p_1, p_2 \ldots p_n)}{C(p_1, p_2 \ldots p_n)} = f(p_1, p_2 \ldots p_n)
  \]

  \[
  V_{\text{max}} = \sup(f(p_1, p_2 \ldots p_n))
  \]
Metric space*

“Design Space” (Parameter space)

“Performance Space”

“Metric Space” (Value space)

*Not quite the same as (but not completely unrelated to) *Sur quelques points du calcul fonctionnel
“Design Space” (Parameter space)

“Performance Space”

“Metric Space” (Value space)

Permissible configurations

*Not quite the same as (but not completely unrelated to) *Sur quelques points du calcul fonctionnel*
Design optimisation

- For many systems, we can explicitly solve for the optimal design point
  - Estimate is only as good as your value function
  - Gradient of the value function is the design parameter “sensitivity”
Finding a value function

• How do we encode the utility of a design?
  – Highly subjective: what does “best” mean?
• Many tools for thinking about utility
  – Multi-criteria decision analysis
  – Pairwise comparison
  – Decision matrix method
  – Management by objectives

There is a whole field of “value engineering”
A quick example

- **Pair-wise decision matrix:**

<table>
<thead>
<tr>
<th></th>
<th>Safe</th>
<th>Low cost</th>
<th>Reusable</th>
<th>Easy to build</th>
<th>Payload capacity</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Safe</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>4</td>
</tr>
<tr>
<td>Low cost</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Reusable</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>Easy to build</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Payload capacity</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td>3</td>
</tr>
</tbody>
</table>

Prioritise by score
“D” for “X”

• How to choose a value function?
  – Design for performance
  – Design for manufacture
  – Design for reliability
  – Design for sustainability
  – Design for cost
  – Design for marketability
  – Design for obsolescence

Increasing cynicism
Of course...

- It is relatively rare that a single value function can fully capture the complex give and take of a real-world design problem
  - Uncertain system constraints/assumptions
  - Uncertain system parameters
  - Uncertain system specifications (!)
  - Mutually exclusive goals
  - Conflicting agendas
  - Conflicting personalities
The most important truth in your degree

Engineering is the art of the trade-off
Methodological approaches

• Ok, that’s great – but how do we do this trade-off thing, exactly?
  – Lots of different ways!
  – Quite likely as many design processes as there are design engineers

• Here are just a few popular design process methodologies…
Alphabet soup

• LAST: Look, Ask, Simplify, Try
• ETC: Express, Test, Cycle
• PDP: “Product Design Process”
  – Define, Model, Implement, Verify, Support
• “Waterfall Model”
  – Requirements, Design, Implement, Verify, Maintain

(And many, many, many more – each more buzzwordy, cliché and feel-good managerial-speak than the last)
Cyclic vs linear models

Model → Define → Support → Implement → Verify

Requirements → Design → Implement → Verify → Maintain

Express → Test

Look → Ask → Simplify → Try
So which do you pick?

Every project is different

Slavish adherence to rigid procedure will (probably) just waste your time

Find what process works for you (and your company)
The common threads

1. Work out what to do
   – Specifications; the design brief – be precise
   – Understand the real constraints

2. Find a solution*
   – Iterate until you do

3. Make sure it works
   – Modelling, validation, testing
   – Critically evaluate ideas

*Wasn’t that the problem to begin with??
The synthesis step

• Constitution + Optimisation = Synthesis

The messy, complicated, creative, intuitive, frustrating, marvellous, deep, ineffable, often iterative intellectual challenge that lies at the heart of all brilliant engineering solutions

Art, not science: Anyone who claims they can teach you to do this is sadly misguided
| 74. | **Take the Ferry across the lake.** | ![Image](image1.png) | ![Image](image2.png) | Go 2.8 mi |
| 75. | **Climb the Hill toward Hangman's Ridge, avoiding any mountain lions.** | | | Up 1.172 ft |
| 76. | **When you reach an old barn, go around back, knock on the second door, and ask for Charlie.** | | | Go 52 ft |
| 77. | **Tell Charlie the dancing stones are restless. He will give you his van.** | | | Careful |
| 78. | **Take Charlie's van down Old Mine Road. Do not wake the straw man.** | | | Go 11 mi |
| 79. | **Turn left on Comstock. When you feel the blood chill in your veins, stop the van and get out.** | | | Go 3.2 mi |
| 80. | **Stand very still. Exits are north, south, and east, but are blocked by a spectral wolf.** | | | Go 0 ft |
| 81. | **The spectral wolf fears only fire. The Google Maps team can no longer help you, but if you master the wolf, he will guide you. Godspeed.** | | | Go ?? mi |
Paul’s philosophy

Here are a collection of words that embody my own personal design philosophy

You may find them helpful.
They are not for everyone.
Philosophy of scope

• Understand the problem (work out what to do)
  – Specification
    • State with precision the problem to be solved
  – Requirements
    • Deconstruct the problem into a set of parameters/metrics the solution must satisfy
  – Background research
    • Identify gaps in your knowledge about the problem
    • Identify gaps in capability needed to solve the problem
    • Experiments where needed to gain information
Philosophy of synthesis

• The Synthesis Step (find a solution)
  – Ideate potential solutions (brainstorming)
  – Critically assess potential solutions (debate)
    • Constructively attack all ideas on merits
    • Promote resilient candidates, cull falsified candidates
  <Special guest appearance by "Design for X" thinking process>
    • Iterate as needed
  – Test candidates (analysis)
    • Demonstrate scientifically that the candidate will successfully solve the problem
    • Math, simulation, small-scale testing as appropriate to gain confidence that the candidate will work
    • Cull falsified candidates
  – Iterate as needed
  – Choose fittest candidate solution to implement
Philosophy of ideation

• Vigorously debate and acid-test all ideas
  – No sacred cows: nothing passes untested
  – Be prepared to support your opinions with facts
• Idea Thunderdome (aka “Conceptual Darwinism”)
  – Two ideas enter, one idea leaves
• Don’t be afraid of maths
  – It won’t bite, and is a powerful ally once tamed

Some people are not comfortable voicing opinions or making themselves heard, but sometimes you need to stand up for your ideas! Solid analysis is your sword and your shield.
Philosophy of execution

• Implementing the solution is not actually a design step
  – But thinking about how the solution will be implemented is!

• Validate the solution
  – Testing is a lifestyle
    • Material test
    • Batch test
    • Spot test
    • Subsystem test
    • Integration test
    • All-up test
  – Critically assess real-world performance viz requirements
  – (Optional: Refine solution)
Recommended reading

• “The Existential Pleasures of Engineering”
  – Samuel C. Florman

• “The Design of Everyday Things”
  – Donald A. Norman

• “Materials and Design”
  – M. Ashby and K. Johnson
Questions?
Design analysis – Part I

Due March 13th

Friday, OMG!
Submit via Platypus, OMG!
Massive wall of ECP text

The objective of the design analysis is to give the student experience in systematically deconstructing a design problem, and in critically assessing proposed solutions. The student must provide a succinct analysis of the engineering challenges embodied in the project task, and propose a feasible solution to be undertaken in resolving them. The student will demonstrate that they understand the underlying engineering problem, its scope, requirements and constraints. Simple restatement of the design task is not sufficient - the student must systematically break down the problem and give insights into the substance of the design tradeoffs. The candidate solution must represent a convincing, achievable approach that can be implemented within the constraints and timeline of the course. The design analysis and proposed design are together to be no more than two A4 pages.
Design analysis – Part I

Three key things you have to do:

• Break down the problem into scope, requirements and constraints.

• Identify and discuss key underlying engineering design challenges.

• Present a candidate solution, and explain how your approach will overcome them.
Not a lot of work, but…

• Just two pages, but that’s not a lot of space
  – So reaaaally think it through
  – Make every word count

• Your peers will be assessing your work:
  – Deconstruction and understanding
  – Quality of work
  – Feasibility of analysis

  Target those areas accordingly
A word from our sponsor…

UQ Racing Electric Vehicle

Now recruiting 4th year students for thesis topics
Tune-in next time for...

Professional Engineering Topics

or

“Stuff they should have taught you at university, but didn’t”

Fun fact: Thirty two monkeys have been launched as part of various space programs – most recently by Iran on 31st January 2013.

Nineteen survived.