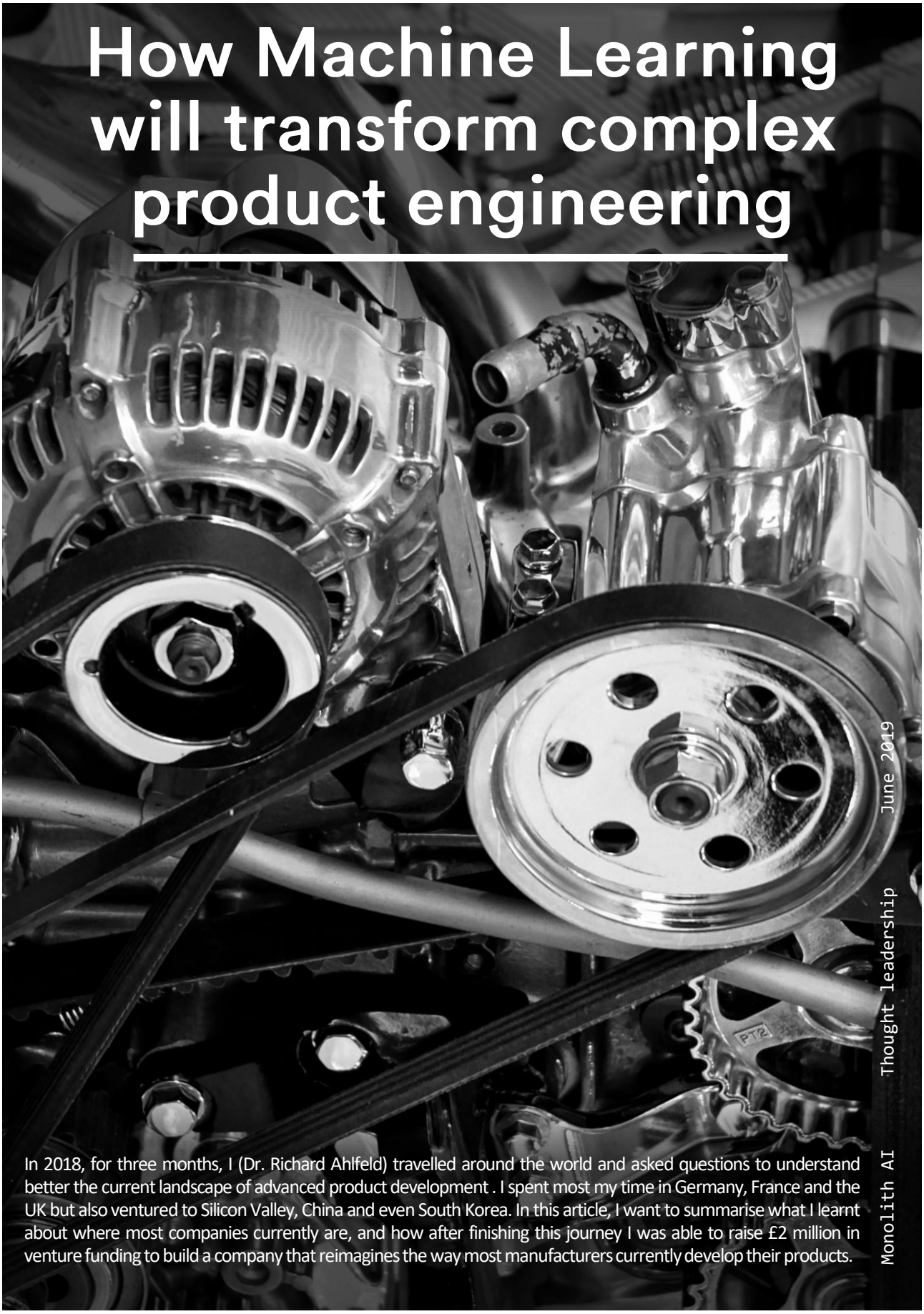
How Machine Learning will transform complex product engineering



In 2018, for three months, I (Dr. Richard Ahlfeld) travelled around the world and asked questions to understand better the current landscape of advanced product development . I spent most my time in Germany, France and the UK but also ventured to Silicon Valley, China and even South Korea. In this article, I want to summarise what I learnt about where most companies currently are, and how after finishing this journey I was able to raise £2 million in venture funding to build a company that reimagines the way most manufacturers currently develop their products.

Monolith AI Thought leadership June 2019

Why I wrote this article

In 2018, I was provided with a curious new opportunity. I won a Techcelerate grant: £30,000 that could only be used for travel and talking to industry experts. It could not be spent on scientific research, software development, product development, or any form of engineering — just on chatting with experts to understand better the current landscape of advanced product development and the state of the art in machine learning.

So for three months, I travelled around the world and asked questions. I spent most my time in Germany, France and the UK but also ventured to Silicon Valley, China and even South Korea. In this article, I want to summarise what I learnt about where most companies currently are, and how after finishing this journey I was able to raise £2 million in venture funding to build a company that reimagines the way most manufacturers currently develop their products: Monolith AI.

Lessons learnt

Developing a complex product — a car, an engine, an aeroplane, or even a lawnmower — inevitably takes a lot of time and costs a lot of money. Most importantly for this article, however, it also creates the most valuable data that engineering companies possess. So it's a bit surprising, at first glance, that the first lesson I learnt was that most digital innovation incentives have ignored this goldmine of data. Instead, they have focussed on logistics, customer analytics, and predictive maintenance or connected devices. That makes sense, of course. These areas generally provide simple, tabulated big data that can be ingested and learnt from.

Advanced Research & Development, on the other hand, uses complicated physical equations and simulation tools that solve partial differential equations, and collect generally seriously noisy and biased sensor data. So, unlike in the other areas, **it's quite hard to find machine learning tools that work out of the box for the kind of data types that are created by R&D departments**. In addition, advanced engineers are generally fairly clever people, so nobody felt the urgent need to explore alternative ways of building products. Sometimes, these experts have even experimented with machine learning themselves and got massively disappointed that they really are not the promising new magic tool that they are sometimes propped up to be. Clever automatically found mathematical models and neural networks have been used since the 70s in controllers, filters, optimisation, and so on.

The first response I usually got from advanced engineers is: oh great, another machine learning expert who is going to say if you give me all your data, I can solve all your problems. There is a reason why we aren't doing this already and it can be summarised in one simple sentence:

"I like sorting my data“
said no engineer ever.

Collecting good data is expensive. Running really informative simulations takes a long time and performing accurate experiments is generally quite expensive. As a result, R&D departments collect as little data as possible to understand a problem. Because of this, they believe that they do not have enough data to learn from. This is where I often found them to be wrong.

In 2016, I was an invited researcher at the Stanford Center of Turbulence Research in California, where I experimented with using surrogate models and machine learning models to better understand flow physics. While I don't think the research I did back then was of great importance, I learnt two very interesting concepts.

1. All engineering efforts are essentially a journey to learn a functional relationship by analysing data — just like all machine learning problems

2. Machine learning applied to physical problems can be significantly more fruitful in engineering than it could ever be in finance or insurance. Why? Because engineering is applying the laws of physics. Thus, there are always underlying physical relationships in all engineering data

Put together, these two points created an entirely new image of engineering and what engineering companies do in my mind. Let's start with the first point.

Whenever somebody performs an experiment, tests a prototype or collects sensor data, they are doing so to understand a system. The human mind usually understands a technical system by understanding one parameter at a time. Even if you follow refined variational analysis methods such as Taguchi's method, you end up with a lot of testing to do. Here's a very simple example. Let's say you want to set the temperature in your electric shower. You can modify the flow velocity and the power of your heating. To get the right temperature, you'll first play with the flow velocity and then play with the heating until you get the temperature you want. This will take you a couple of minutes to do.

If, however, let's say you could ask Alexa or Siri to help you, they would tell you: Set the flow velocity to 75% and the heating to 80% first and then to 40% and 65% second and 85% and 90% third. Then tell me how you feel. Based on these three points they can determine a function and solve an optimisation problem to find your perfect water temperature and flow velocity. It's simple math.

Now imagine applying this same concept to a race car company that spends millions testing their new cars in track tests, or turbomachinery companies that spend tens of millions learning how to calibrate their engines. Algorithms can learn from random data, humans cannot. Engineering is full of problems where this kind of algorithmic engineering can literally save up to 98% of the time spent on trying to understand something.

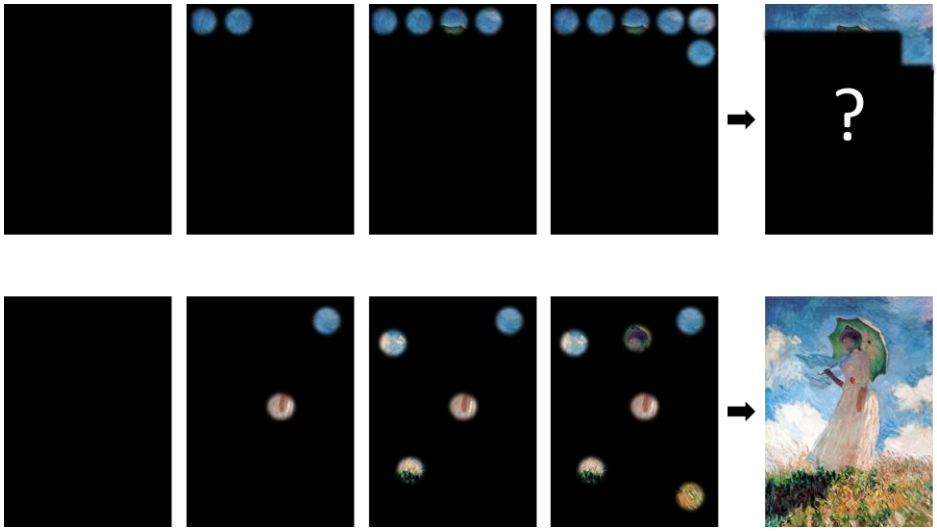
Even though the big automotive companies spend billions using the most recent advances in deep learning in their autonomous cars, none of them uses this simple trick to accelerate their product development and testing time. I found the same goes for turbomachinery manufacturers, packaging manufacturers, defence companies, medical equipment manufacturers, and so on.

Now to the second big lesson learnt.

All engineering data contains physical patterns. It's a simple fact but I'll repeat it anyway. Now, the curious thing is that if you ask an engineer if they carry over data from, let's say, the 2017 car to the 2018 car, they'll say: oh no. That doesn't work. The physics is too complicated and non-linear to make sense. They are right in many cases. However, let's say you build a model to optimise the fuel efficiency of your combustion engine for 2017. In 2018, you will build another model. Since both designs are different, the engineers in both cases will collect the exact same data: run the same simulations, perform the same tests. If you would have trained a Gaussian Process model on the final results from last year, and perfectly tuned its basis functions to describe the results in a most physical sense, and you reused this model this year as what machine learning experts call 'your prior' assumptions and consider this year's learning as an extension of last year's, I guarantee you will find the answers you're looking for with around 70% less testing than before. Using the Monolith approach with clever priors, we have managed to find system responses for turbomachinery examples with as little as 12 tests, down from 100.

To conclude this article, I would like to tell you one of my favourite anecdotes to explain why machine learning could drastically accelerate product development beyond what engineering companies currently do.

Imagine I put you in a huge empty hall with nothing but a box of matchsticks. In front of you, is a 100 square meter big painting on the wall but you don't know what it looks like. Training in logical thinking you will probably do a grid search. You will light a match stick every two meters and start building a picture. If you just use one matchstick per square meter you'll need 100 to figure out what's going on. If you used algorithmic engineering, you would start with 5 random locations. And then refine your search based on what you find. Let's say the top three are all blue sky. It makes no sense to search further. So instead you'll sample more in the bottom regions. You find water lilies. I assume you can create a really good approximation of the image within 20 samples — an 80% saving. Last but not least, let's say you use algorithmic engineering with prior knowledge of having trained your search algorithm on all major large artworks in the world. As soon as you find the first impressionist water lilies who will know it's the Orangerie in Paris and the painting is by Claude Monet!



Engineering design is often like trying to figure out what a painting looks like if you can't see the whole picture. Inference and learning can achieve the same results as a grid search much faster!

Let's get back to the current stage of engineering. In all the engineering departments, I visited, whether it was Defence, Aerospace, Automotive, or Packaging I found at least one high-value application where huge amounts of resources were wasted trying to understand a complex problem using manual search methods instead of algorithmic ones.

For more information
visit monolithai.com