

Hybride und energieeffiziente Antriebe für mobile Arbeitsmaschinen

8. Fachtagung
Karlsruhe, 23. Februar 2021

Herausgegeben von

Institutsteil Mobile Arbeitsmaschinen (Mobima),
Karlsruher Institut für Technologie (KIT)

Prof. Dr.-Ing. Marcus Geimer

Verband Deutscher Maschinen- und Anlagenbau (VDMA)

Dipl.-Ing. Peter-Michael Synek

Impressum



Karlsruher Institut für Technologie (KIT)
KIT Scientific Publishing
Straße am Forum 2
D-76131 Karlsruhe

KIT Scientific Publishing is a registered trademark
of Karlsruhe Institute of Technology.
Reprint using the book cover is not allowed.

www.ksp.kit.edu



*This document – excluding the cover, pictures and graphs – is licensed
under a Creative Commons Attribution-Share Alike 4.0 International License
(CC BY-SA 4.0): <https://creativecommons.org/licenses/by-sa/4.0/deed.en>*



*The cover page is licensed under a Creative Commons
Attribution-No Derivatives 4.0 International License (CC BY-ND 4.0):
<https://creativecommons.org/licenses/by-nd/4.0/deed.en>*

Print on Demand 2021 – Gedruckt auf FSC-zertifiziertem Papier

ISSN 1869-6058 (Schriftenreihe)

ISSN 2510-7275 (Tagungsband)

ISBN 978-3-7315-1071-0

DOI 10.5445/KSP/1000127640

Die Orchestrierung digitaler Zwillinge für Industrie 4.0

Heiko Baum¹; Benedikt Müller¹ und Oliver Breuer¹

¹ FLUIDON Gesellschaft für Fluidtechnik mbH, Aachen, Deutschland
E-Mail: heiko.baum@fluidon.com; Tel.: (+49) 241 9609260

Kurzfassung

Die Frage: Was sind die „Antriebslösungen unter Einbezug von Industrie 4.0 – Enabler für hybride und energieeffiziente Antriebe mobiler Arbeitsmaschinen und Anbaugeräte“ führt früher oder später zu der Antwort: „Der digitale Zwilling ist ein entscheidender Enabler von Industrie 4.0“. Wie aber kann ein digitaler Zwilling sein Potenzial als Treiber für die Industrie 4.0 ausschöpfen und den Entwickler bei der täglichen Arbeit unterstützen? Die Lösung lautet: „Der digitale Zwilling bedarf einer geeigneten Orchestrierung“, denn nur dann profitiert der Entwickler davon, dass er die virtuelle Repräsentanz seiner Maschine unter einer Vielzahl von Randbedingungen simulieren kann, um zu sehen, wie diese sich verhält.

Der Beitrag verdeutlicht am Beispiel der Druckschwingungsanalyse eines mobil-hydraulischen Antriebs die Anforderungen an die Orchestrierung. Es wird demonstriert, wie unter Nutzung eines vorhandenen digitalen Zwillings ein Workflow individuell aufgesetzt und im Virtual Engineering Lab (VEL) abgebildet wird.

Schlagnworte: Orchestrator, Automatisierung des Simulations-Workflows, digitaler Zwilling, FMU, Python

1 Einleitung

Eine Antwort auf die Frage „wie können Antriebslösungen unter Einbezug von Industrie 4.0 als Enabler für hybride und energieeffiziente Antriebe mobiler Arbeitsmaschinen und Anbaugeräte fungieren“ ist der digitale Zwilling [1]. Der digitale Zwilling beinhaltet ein Simulationsmodell – die virtuelle Repräsentanz der realen Maschine – dessen Potenzial es auszuschöpfen gilt, um den Entwickler bei der täglichen Arbeit zu unterstützen. Hierzu bedarf es allerdings der geeigneten Orchestrierung des digitalen Zwillings, also dem flexiblen Kombinieren mehrerer Arbeitsschritte zu einem Workflow [2]. Nur durch die konsequente Automatisierung dieses Workflows profitiert der Entwickler davon, dass er die virtuelle Repräsentanz seines Systems unter einer Vielzahl von Randbedingungen simulieren kann, um zu sehen, wie diese sich verhält.

Die Frage „was ist eine geeignete Orchestrierung?“ kann allerdings nicht verallgemeinert beantwortet werden. Die geeignete Orchestrierung lässt sich nur fallspezifisch sinnvoll realisieren, denn abhängig von der zu untersuchenden Fragestellung geht ein digitaler Zwilling unterschiedlich stark ins Detail und kann sowohl ein vereinfachtes Datenmodell als auch ein physikalisch modelliertes Multi-Domain-Modell [3] sein. Zur Orchestrierung des Workflows wird daher ein flexibles, idealerweise plattform- und produktoffenes und individuell konfigurierbares Werkzeug benötigt, mit dem der Anwender seinen Workflow definiert.

2 Druckschwingungsanalyse eines hydrostatischen Antriebsstrangs

Die Anforderungen an einen flexiblen Workflow werden am Beispiel eines hydrostatischen Antriebsstrangs (Abbildung 2.1) beschrieben, der in der Regel mindestens eine Fahrpumpe und mehrere Fahrmotoren umfasst. Um den Antriebsstrang im optimalen Wirkungsgradbereich zu betreiben oder um spezielle

3 Der VEL-Workflow

Zur Auslegung des Leitungssystems wird mit dem Virtual Engineering Lab (VEL) ein Workflow aufgesetzt, der mittels Parametervariation eine Vielzahl möglicher Leitungssystemkonfigurationen untersucht. Die Oberfläche des VEL (Abbildung 3.1) stellt dem Anwender verschiedene Module für die Abbildung des Workflows zur Verfügung.

Neben DSHplus-Simulationsmodellen [4] können auch Linux-FMU-Modelle (functional mock-up unit) aus allen FMI-konformen Entwicklungstools importiert werden [5].

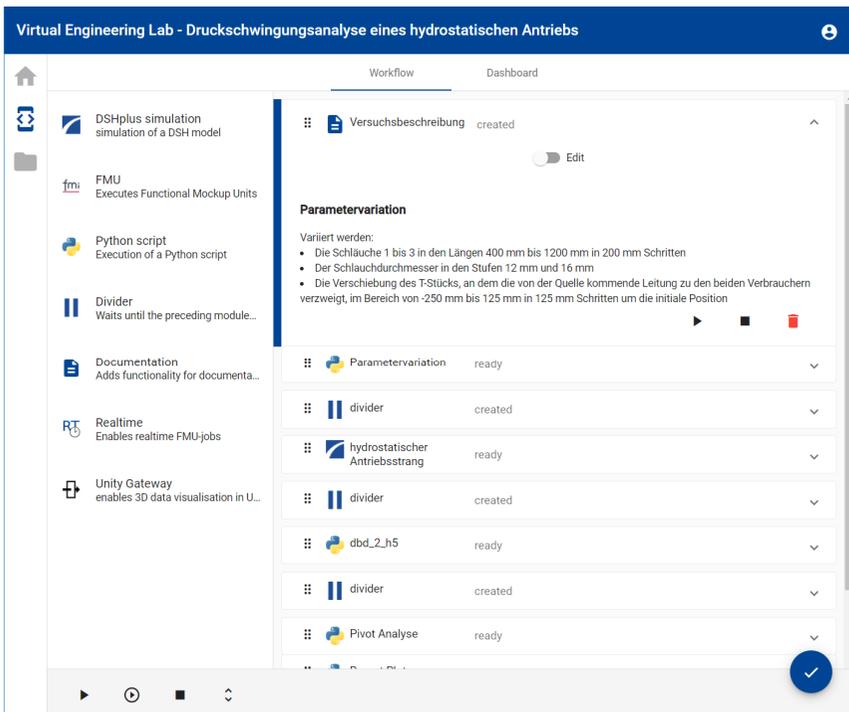


Abbildung 3.1: VEL-Orchestrator mit Beispiel-Workflow zur Druckschwingsanalyse.

Pre-Processing-Arbeitsschritte, um z. B. Parameterstudien durchzuführen oder Trainingsdaten für das Machine Learning zu erzeugen, werden mittels Python-Skript realisiert. Python ist auch das geeignete Werkzeug für das anschließende Post-Processing der Simulationsergebnisse oder von Messdaten und kann für Big-Data-Analysen eingesetzt werden. Das Documentation-Plugin ist dazu gedacht, den Workflow mit Kommentaren, die auch Formeln im LaTeX-Format enthalten können, zu beschreiben. Das VEL lässt sich durch ein RT-Plugin an Feldbusnetzwerke ankoppeln, um den Entwicklern mobiler Arbeitsmaschinen virtuelle Inbetriebnahmetechniken für SiL- und HiL-Anwendungen zur Verfügung zu stellen [6]. Das Plugin Unity-Gateway leitet Daten aus dem VEL in Echtzeit an eine 3D-Visualisierung weiter.

Sowohl die VEL-Benutzeroberfläche mit den Plugins als auch das darunterliegende Software-Framework basieren auf Open-Source-Toolkits und standardisierten Schnittstellendefinitionen, wodurch ein Workflow sehr leicht um neue Bausteine erweitert und an unternehmensspezifische Anforderungen angepasst werden kann.

Im VEL ist die Struktur des Workflows entsprechend des aus der Softwareentwicklung bekannten „Fork-Join Modells“ [7] (Abbildung 3.2) organisiert.

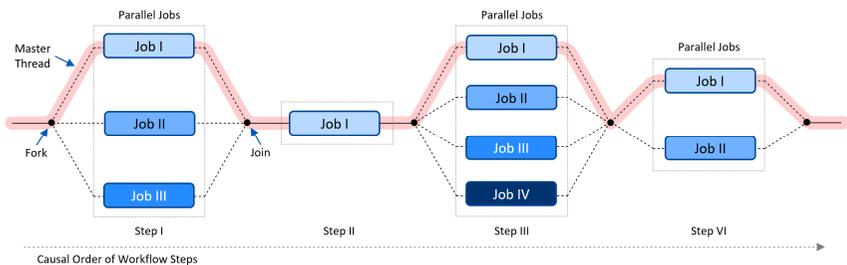


Abbildung 3.2: Verarbeitung von Arbeitspaketen mit dem Fork-Join Modell.

Im Fork-Join Modell werden die Arbeitspakete zunächst entsprechend ihrer kausalen Reihenfolge sortiert und anschließend die Arbeitspakete, die

voneinander unabhängig sind, parallel verarbeitet (Fork). Der Workflow wird erst dann wiedervereinigt (Join), wenn Ergebnisse aus parallelen Arbeitspaketen als Eingänge für nachfolgende Arbeitspakete benötigt werden.

Das VEL verarbeitet die Jobs eines Workflows standardmäßig immer parallel. Mittels eines Trenners (Divider) kann der Anwender aber das Wiedervereinigen des Workflows erzwingen, sodass die in Abbildung 3.2 präsentierte Fork-Join-Struktur entsteht.

4 Parameterstudie mit dem VEL

Zu Beginn einer Parameterstudie gilt es die Grenzen zu definieren, in denen die Bauteilparameter variiert werden, und es muss festgelegt werden, wie die Ergebnisse zu bewerten sind. Abbildung 4.1 zeigt einen Ausschnitt des für die Parameterstudie vorbereiteten Simulationsmodells des Antriebsstrangs.

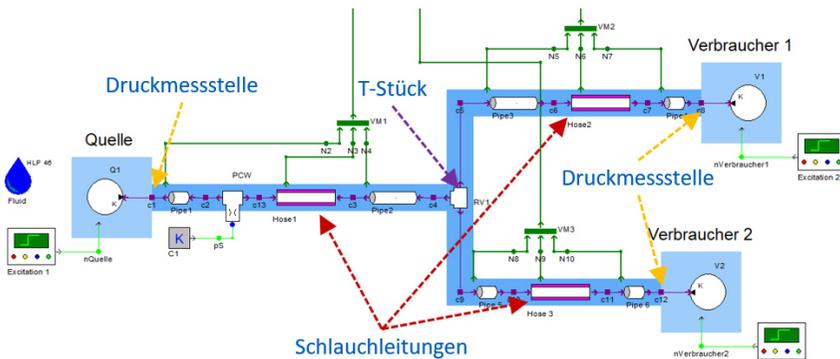


Abbildung 4.1: Simulationsmodell des hydrostatischen Antriebsstrangs.

Aus Bauraumgründen müssen die Leitungslängen des Antriebsstrangs konstant bleiben. Allerdings können die Längen der Schlauchleitungen 1 bis 3 variable zwischen 400 mm und 1000 mm gestaltet werden. Durch Formelbeziehungen

in den Bauteilen werden die Längen der verbleibenden Stahlleitungen automatisch entsprechend angepasst. Als weitere Variationsmöglichkeiten stehen Schlauchleitungen mit den Durchmessern 12 mm und 16 mm zur Auswahl und die Positionierung des T-Stücks, an dem die von der Quelle kommende Leitung zu den beiden Verbrauchern verzweigt, kann von -250 mm bis +125 mm aus der Ausgangsposition verschoben werden. Abbildung 4.2 zeigt die Ansicht des für die Parametervariation zuständigen Python-Skripts im VEL.

Virtual Engineering Lab - Druckschwingungsanalyse eines hydrostatischen Antriebs

Workflow Dashboard

Versuchsbeschreibung created

Parametervariation ready

Home Graphics

```

23  ## Change in length as an array
24  laenge_1 = np.arange(400.0, 1000.01, 200.0)
25  laenge_2 = np.arange(400.0, 1000.01, 200.0)
26  laenge_3 = np.arange(400.0, 1000.01, 200.0)
27  ## Change in diameter as an array
28  durchmesser = np.arange(12.0, 16.01, 4.0)
29  ## change in position as an array
30  t_pos = np.arange(-250.0, 125.01, 125.0)
31
32  # build the DOE data frame
33  #-- Skript -----
34  t0 = time.time()
35  print('%s:if s: '%(time.time() - t0) + 'build

```

0.0 s: build the DOE data frame

	Par1	Par2	Par3	Par4	Par5
0	400.0	400.0	400.0	12.0	-250.0
1	600.0	400.0	400.0	12.0	-250.0
2	800.0	400.0	400.0	12.0	-250.0
3	1000.0	400.0	400.0	12.0	-250.0
4	400.0	600.0	400.0	12.0	-250.0
...
507	1000.0	800.0	1000.0	16.0	125.0
508	400.0	1000.0	1000.0	16.0	125.0
509	600.0	1000.0	1000.0	16.0	125.0
510	800.0	1000.0	1000.0	16.0	125.0
511	1000.0	1000.0	1000.0	16.0	125.0

Datei auswählen Keine Dat... ausgewählt

Submit

divider created

Abbildung 4.2: Python-Skript zur Parametervariation mit vollfaktoriellem Versuchsplan.

Im Python-Skripts dienen die Zeilen 24 bis 30 zur Eingabe des zu variierenden Parameterbereichs. Das Skript erzeugt hieraus einen vollfaktoriellen Versuchsplan [8] mit 512 Parametersätzen und speichert diese direkt im Verzeichnis des Simulationsmodells, das jetzt, entsprechend der Workflow-Reihenfolge aus

Abbildung 3.1, mit der Berechnung beginnt. Vom VEL-Plugin für die Simulationsrechnung werden die voneinander unabhängigen Parametersätze parallel berechnet, wodurch die volle Leistung der Rechnerhardware ausgenutzt wird.

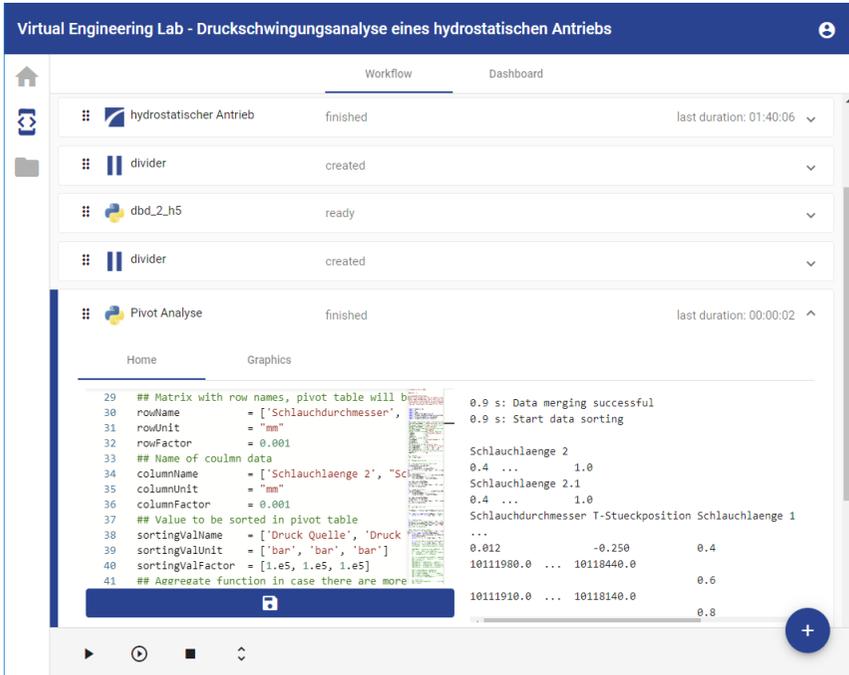


Abbildung 4.3: Python-Skript zur Pivot-Analyse.

Die detaillierten Ergebnisse der 512 Simulationen sind für den Entwickler zunächst allerdings von untergeordnetem Interesse. Primär möchte er durch die Parametervariation absichern, dass es bei der vom ihm gewählten Leitungskonfiguration zu keiner störenden Resonanzsituation im Antriebsstrang kommt, und außerdem erfahren, welche der möglichen Leitungskonfiguration die niedrigsten Druckpulsationsamplituden aufweist. Als Kriterium für die Bewertung dienen die Drücke an den in Abbildung 4.1 markierten Druckmessstellen. Die

Extraktion der Druckwerte aus den Simulationsergebnissen übernimmt wiederum ein Python-Skript (Abbildung 4.3).

Im linken Teil des Python-Skripts sind die Eingaben zu sehen, die festlegen, in welcher Formatierung die selektierten Ergebnisse als Pivot-Tabelle ausgegeben werden. Der rechte Teil zeigt einen Teilausschnitt der Pivot-Tabelle, die zur Weiterverarbeitung als Microsoft-Excel-Datei gespeichert wird.

Abbildung 4.4 präsentiert die Pivot-Tabelle der Druckamplituden an Verbraucher 2. Durch eine farbliche Kennzeichnung ist direkt zu erkennen, welche Leitungssystemkonfiguration höhere und welche niedrigere Druckwerte verursacht. Die Zuordnung der Druckwerte zu einer bestimmten Leitungssystemkonfiguration erfolgt über die Zeilen- und Spaltensortierung der Tabelle. Die Druckwerte sind zeilenweise zunächst nach den beiden Schlauchdurchmessern sortiert, dann folgt die Sortierung nach der T-Stückposition und schließlich die Sortierung nach der Schlauchlänge 1. Die Spalten sind zuerst nach der Schlauchlänge 2 und dann nach der Schlauchlänge 3 sortiert.

		0,4				0,6				0,8				1				
Schlauchdurchmesser	T-Stückposition	Schlauchlänge 1				Schlauchlänge 2				Schlauchlänge 3				Schlauchlänge 4				
		0,4	0,6	0,8	1	0,4	0,6	0,8	1	0,4	0,6	0,8	1	0,4	0,6	0,8	1	
0,012	0,25	0,4	10,79	8,11	7,98	7,05	9,53	8,86	8,06	8,77	11,17	9,61	9,29	9,68	11,45	12,01	11,61	8,43
		0,6	11,25	8,55	7,93	7,26	9,94	8,98	8,28	8,01	11,47	9,88	9,29	9,02	11,04	11,97	11,40	8,93
		0,8	11,62	8,91	8,07	7,75	10,45	9,18	8,33	7,70	11,59	10,04	9,37	8,69	12,22	11,74	11,24	9,22
		1	11,77	9,28	8,18	9,06	10,90	9,49	8,81	8,40	11,52	10,44	9,97	9,31	12,94	12,12	11,87	9,15
	-0,125	0,4	9,29	8,49	7,22	6,84	9,88	8,36	7,43	6,66	10,42	9,08	8,06	7,42	10,87	9,61	8,61	8,45
		0,6	9,55	8,46	7,30	7,15	10,02	8,53	7,83	6,77	10,45	9,67	8,42	7,68	11,25	9,76	8,80	8,30
		0,8	9,83	8,46	7,46	7,29	10,09	8,67	8,12	7,04	10,99	9,89	8,66	7,50	11,46	9,87	8,74	8,34
		1	10,02	8,60	7,73	7,58	10,36	8,89	8,06	7,37	11,44	9,75	8,51	7,80	11,40	10,09	9,08	8,52
	0	0,4	8,61	7,45	6,44	6,34	8,62	7,56	6,77	6,13	9,59	8,19	7,41	6,58	9,68	8,37	7,55	6,97
		0,6	8,78	7,74	6,83	6,57	9,20	7,91	6,96	6,29	9,86	8,56	7,58	6,72	10,28	8,87	7,63	7,27
		0,8	9,08	7,85	6,94	6,96	9,53	8,12	7,12	6,51	10,10	8,66	7,96	7,06	10,45	9,05	7,78	7,18
		1	9,49	8,21	6,93	7,23	9,70	8,30	7,27	6,48	10,45	8,72	7,87	7,06	10,86	9,68	8,17	7,11
0,125	0,4	7,48	6,38	6,21	6,28	8,08	6,80	6,25	5,96	8,85	7,67	7,53	6,46	8,97	8,02	7,09	6,11	
	0,6	7,96	6,72	6,17	6,62	8,17	7,02	6,38	6,32	8,90	7,97	8,10	6,12	9,51	7,92	7,07	6,46	
	0,8	8,31	7,18	6,54	6,51	8,79	7,25	6,47	6,43	9,69	8,17	8,09	6,74	9,66	8,19	7,27	6,57	
	1	8,33	7,35	6,85	6,85	8,77	7,36	6,58	6,50	9,50	8,09	8,86	6,57	9,93	8,49	7,60	6,74	
0,016	-0,25	0,4	8,15	6,28	5,36	6,24	8,92	6,99	5,45	6,02	9,08	7,27	6,02	5,62	9,76	7,93	7,26	5,69
		0,6	8,48	6,88	5,28	6,24	9,36	7,18	5,66	5,93	10,05	7,59	6,20	5,62	10,43	7,93	6,61	5,16
		0,8	8,31	6,96	5,24	5,37	9,71	7,30	5,89	5,26	9,81	7,94	6,29	6,20	10,04	8,45	6,68	5,18
		1	8,95	7,04	5,38	5,22	10,10	7,14	5,74	6,37	10,71	7,82	6,16	6,56	10,59	8,28	7,09	6,01
	-0,125	0,4	7,59	5,80	4,90	6,53	7,93	6,30	5,22	6,23	8,59	6,37	5,91	6,00	9,48	7,11	6,43	7,20
		0,6	8,05	6,00	5,06	6,54	8,20	6,52	5,46	6,40	9,09	7,11	6,17	6,07	9,65	7,93	6,29	6,85
		0,8	8,15	6,37	5,48	6,39	8,44	7,20	5,89	6,31	9,35	7,54	6,92	5,88	10,59	7,75	6,54	7,03
		1	8,41	6,75	5,35	6,81	8,87	7,04	5,72	6,47	9,78	7,47	6,44	5,67	10,63	7,76	6,32	6,53
	0	0,4	7,12	5,39	5,27	6,23	7,50	5,72	5,14	5,95	8,13	6,40	6,17	5,78	8,45	6,61	5,93	5,94
		0,6	7,21	5,39	5,08	5,95	7,60	5,89	4,68	5,96	8,41	6,54	6,61	5,86	9,28	7,23	6,06	6,03
		0,8	7,72	5,85	5,66	5,56	8,19	6,12	5,04	5,95	8,93	6,94	6,95	5,16	9,33	7,35	6,09	5,98
		1	7,85	5,86	5,44	5,88	8,09	6,44	5,22	5,61	9,04	7,24	7,39	6,63	9,32	7,44	6,29	6,34
0,125	0,4	6,39	4,72	4,82	5,73	7,14	5,61	4,38	5,13	7,83	6,09	6,38	4,92	8,34	6,43	5,14	4,53	
	0,6	6,98	4,83	4,39	5,01	7,24	5,77	4,52	4,94	8,04	6,34	5,46	4,97	8,47	6,70	5,62	5,96	
	0,8	7,07	5,10	4,71	4,81	7,55	5,86	4,42	4,93	8,12	6,23	5,60	5,76	9,48	7,36	6,12	6,28	
	1	7,49	5,21	5,18	5,04	7,76	6,01	4,56	5,00	8,35	6,66	5,31	5,80	8,86	6,87	5,60	6,31	

Abbildung 4.4: Pivot-Tabelle des Drucks an Verbrauchers 2.

Aus Abbildung 4.4 geht hervor, dass die Konfiguration mit den Schlauchlängen 1 = 1000 mm, 2 = 1000 mm, 3 = 400 mm, dem Durchmesser 12 mm und der T-Stückposition -250 mm mit fast 13 bar die höchste Druckpulsationsamplitude aufweist und daher eher ungeeignet ist, wohingegen z. B. die Konfiguration mit den Schlauchlängen 1 = 600 mm, 2 = 400 mm, 3 = 800 mm, dem Durchmesser 16 mm und der T-Stückposition +125 mm mit ca. 4,4 bar eine niedrige Druckpulsationsamplitude aufweist und geeignet erscheint. Allerdings muss dieser Befund noch mit den Druckpulsationsamplituden an den beiden anderen Messstellen gegengeprüft werden, um abschließend zu bewerten, ob die Konfiguration tatsächlich das beste Gesamtergebnis liefert.

5 Detailanalyse der Ergebnisse

Selbstverständlich kann der Entwickler jederzeit auch genauer auf die Simulationsergebnisse schauen. Hierzu wird der Workflow bedarfsweise um weitere Python-Skripte ergänzt. Abbildung 5.1 zeigt einen Reportplot mit den Drehzahl- und Drucksignalen der als brauchbar identifizierten Konfiguration.

In den Zeitsignalen der Drücke ist sichtbar, dass es im Leitungssystem zu Resonanzsituationen kommt, die die hohen Druckpulsationsamplituden verursachen. Was noch fehlt ist das Verständnis, welche Drehzahlen und welcher Hydraulostat hierfür verantwortlich sind und welche Schwingungssituation daraus resultiert.

Mit der Spektrogramm-Analyse der Drucksignale wird ermittelt, bei welchen Drehzahlen es zu Resonanzen im Leitungssystem des Antriebsstrangs kommt. Abbildung 5.2 zeigt die Spektrogramm-Analyse des Drucksignals am Verbraucher 2.

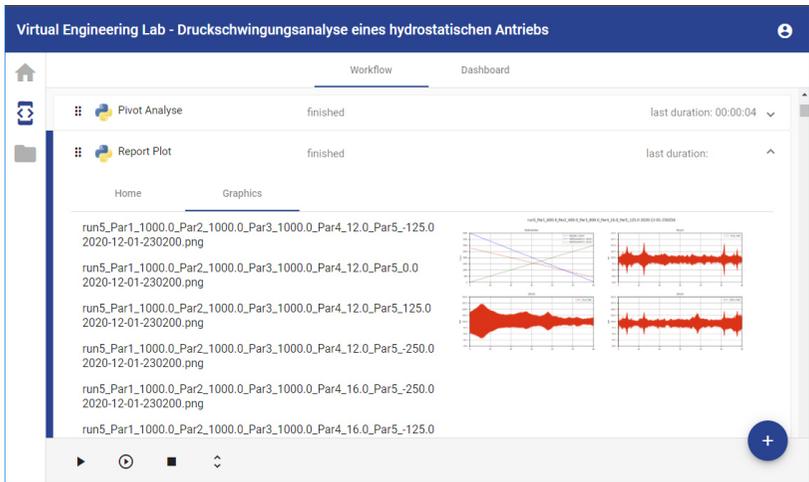


Abbildung 5.1: Python-Skript zur Kurvendarstellung.

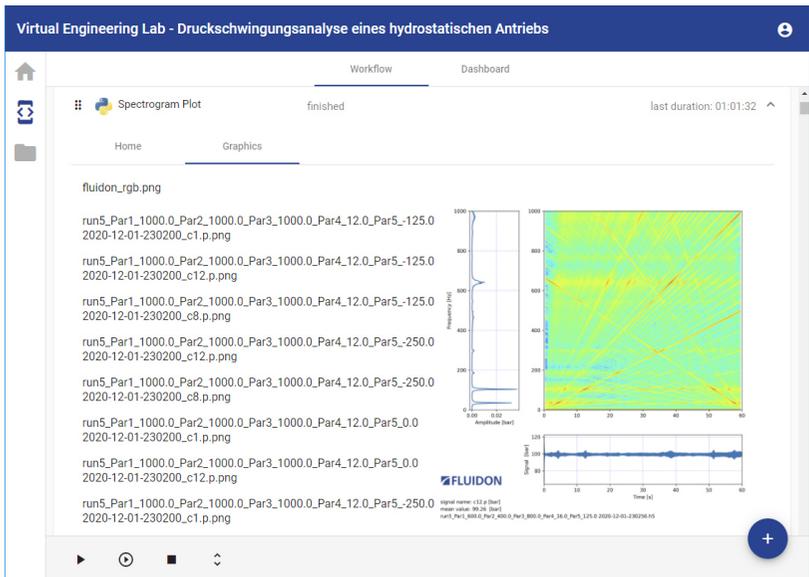


Abbildung 5.2: Python-Skript zur Spektrogramm-Analyse.

Es ist deutlich zu erkennen, dass es bei ca. 50 Hz und ca. 110 Hz eine Systemresonanz mit hohen Amplituden und bei ca. 620 Hz eine sehr breitbandige Resonanz gibt. Welche Druckschwingungsformen des Leitungssystem hiermit einhergehen, wird durch einen Druckvektorplot identifiziert, bei dem die Druckschwingungssituation entlang der Leitungssachse visualisiert wird. Abbildung 5.3 präsentiert das hierzu verwendete Python-Skript. Für eine detaillierte Beschreibung und Interpretation von Spektrogrammen und Druckvektorplots sei an dieser Stelle auf [9] verwiesen.

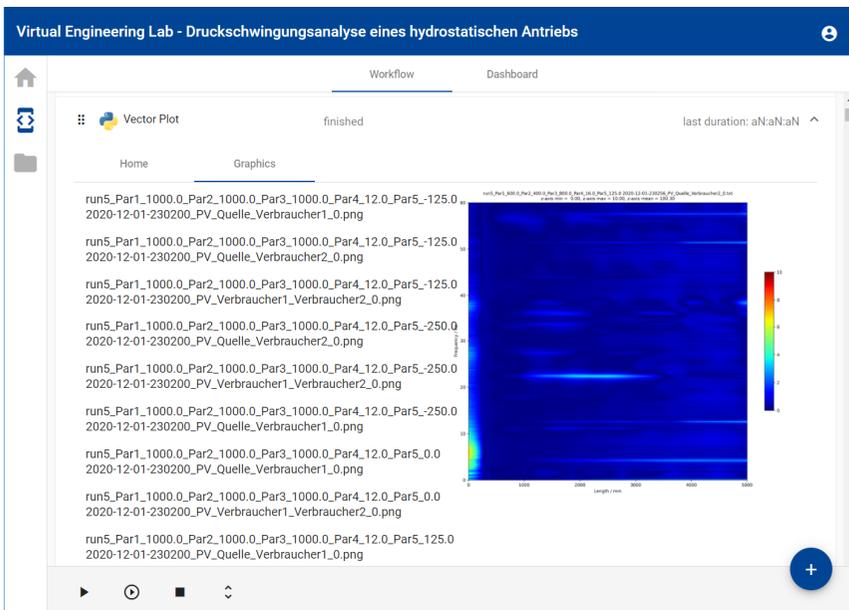


Abbildung 5.3: Python-Skript zur Erstellung von Druckvektorplots.

6 Zusammenfassung

Am Beispiel eines hydrostatischen Antriebs wurde demonstriert, wie mit dem VEL ein Workflow orchestriert wird, um die konstruktiv möglichen Varianten

eines Leitungssystems hinsichtlich ihres Druckpulsationsverhaltens zu überprüfen und die optimale Variante zu bestimmen. Das VEL ist jedoch nicht auf fluidtechnische Fragestellungen limitiert, sondern ist als Orchestrator für Simulationsworkflows aus beliebigen technischen Anwendungen konzipiert. Mittels FMU-Plugin können hierzu digitale Zwillinge aus unterschiedlichen Simulationswerkzeugen in den Workflow integriert werden. Die flexible Zusammenstellung des Workflows aus vordefinierten Plugins ermöglicht anschließend den Einsatz des VEL für die unterschiedlichsten Fragestellungen.

Literatur

- [1] VDI/ VDE: Simulation und digitaler Zwilling im Anlagenlebenszyklus, VDI-Statusreport, 02.2020
- [2] Orchestrierung, <https://de.wikipedia.org/wiki/Dienstekomposition#Orchestrierung>, zuletzt besucht 11.2020
- [3] Multi-Domain-Modellierung, <https://systemdesign.ch/wiki/Multi-Domain-Modellierung>, zuletzt besucht 11.2020
- [4] DSHplus - Simulation hydraulischer und pneumatischer Systeme im mechatronischen Umfeld, <https://www.fluidon.com/>, zuletzt besucht 11.2020
- [5] FMI-standard - Functional Mock-up Interface for Model Exchange and Co-Simulation, <https://fmi-standard.org>, zuletzt besucht am 11.2020
- [6] VEL - eine modulare virtuelle Inbetriebnahmeumgebung für mobile Arbeitsmaschinen, 11. Kolloquium Mobilhydraulik, 10. September 2020, Karlsruhe, Seite 133 – 146
- [7] Fork-Join Model, https://en.wikipedia.org/wiki/Fork%E2%80%93join_model, zuletzt besucht 11.2020
- [8] Statistische Versuchsplanung, https://de.wikipedia.org/wiki/Statistische_Versuchsplanung, zuletzt besucht 11.2020

- [9] Baum, H.: Druckschwingungsanalyse hydrostatischer Antriebsstränge, O+P, 06-2019, Seite 36 - 41