Dr. Serge ZACHER VERLAG
Stuttgart
D-70188

Automation-Letter Nr. 22
06.10.2016

Hannes Homburger und Daniel Wolf

Springender Ball

Seminararbeit zum Fach “Grundlagen Automation” an der DHBW Stuttgart,
Betreuer Prof. Dr. S. Zacher

“Die dynamische Darstellung von physikalischen Größen eines Prozesses wie Temperatur, Druck, Strom usw. auf einer Benutzerschnittstelle nennt man *Prozessvisualisierung*.

Die Prozessvisualisierung ist auf verschiedenen Plattformen vorstellbar:

- in der Entwicklungsumgebung am Büro-PC
- am Display an den Maschinen
- an einem Bedienpanel

Die Abbildung von Prozessgrößen erfolgt grafisch oder numerisch, als Text oder Tabelle. Über Eingabefelder (z. B. Schalter, Tastatur) kann der Operator in den Prozess eingreifen, um Zustände bzw. Sollwerte zu verändern. Weiterhin kann man die Steuerungselemente am Bildschirm als grafische Objekte in Farbe darstellen.“

S. Zacher, C. Wolmering: *Prozessvisualisierung.*, 2015, Wiesbaden. ISBN 978-3-937638-17-1

Abstract, Urheberrechts- und Haftungshinweis

Die Bewegung eines unter der Gravitationskraft der Erde springenden Balles wurde in MATLAB simulieren und anschließend visualisiert. Andere Kräfte wie die Luftreibung (*Stokes*- und *Newton*-Reibung) wurden aufgrund des beschränkten Zeitfensters der Bearbeitung der Seminararbeit vernachlässigt.

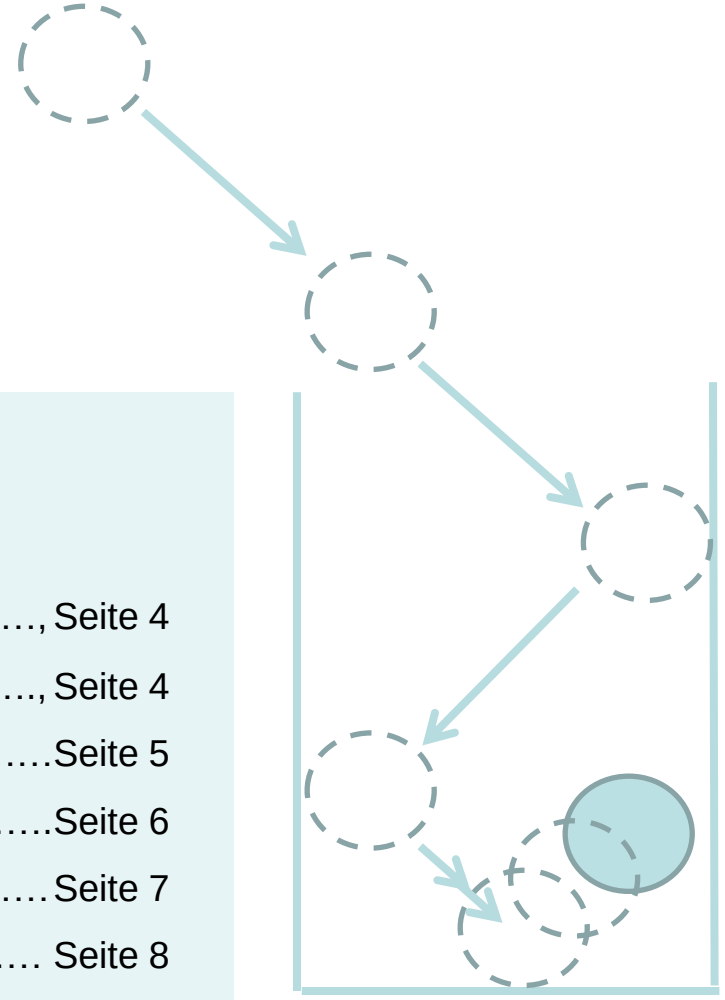
Um eine Abbildung der realen Bewegung des Balles im freien Fall in einem abgeschlossenen Raum darzustellen, wurden die physikalischer Zusammenhänge einer beschleunigten Bewegung analysiert.

Die Bewegung des Balles wurde besteht aus drei Phasen:

- Freier Fall
- Aufprallphase
- Rückprallen

Als Ergebnis entstand ein MATLAB-Skript, mit dem die Bewegung des Balles nahe zu physikalischen Vorgängen simuliert und in 2D-Grafik mit folgenden numerischen Anzeigen des Ballzustanden visualisiert wurde:

- Position
- Geschwindigkeit
- Energie



I N H A L T :

1 Einführung: Visualisieren mit MATLAB

Die Visualisierung eines Prozesses ist grundlegender Bestandteil dessen Automatisierung. MATLAB bietet verschiedene Möglichkeiten diese zu entwerfen.

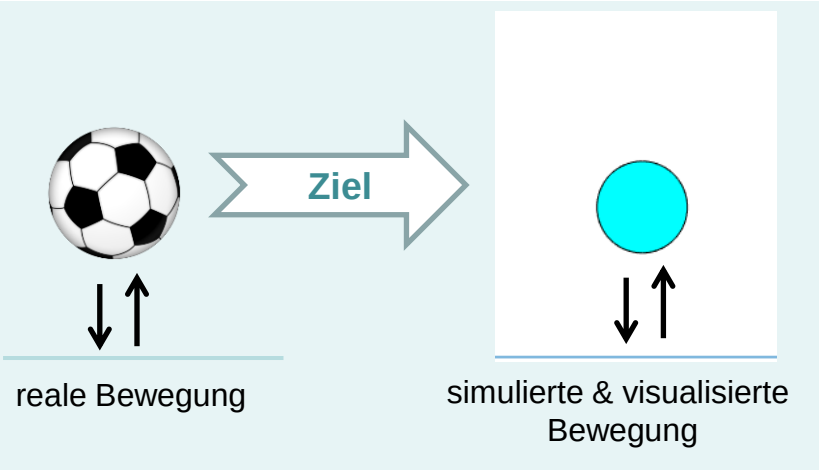
Einerseits ist es möglich Variablen bzw. Variablenfolgen anhand der programmeigenen Funktionen *plot* oder *stem* zu visualisieren. Diese Funktionen erstellen Säulendiagramme, welche beispielsweise den zeitlichen Verlauf einer Prozessvariablen sehr übersichtlich darstellen können. Der Benutzer kann die Visualisierung leicht anhand optionaler Übergabewerte an die Funktionen *plot* und *stem* modifizieren.

Andererseits kann eine **Vektorgrafik** mit dem Befehl *figure* erstellt werden. In diese können Objekte eingebunden werden, um beispielsweise einen Prozess zu visualisieren. Sie besitzen Eigenschaften wie z.B. Farbe oder Größe, welche für die Darstellung maßgeblich sind.
Beispiele für Objekte sind Rechteck (*rectangle*), Linie (*line*) oder Text (*text*).

Zusätzlich gibt es noch die Möglichkeit einer Visualisierung mit **Simulink** – diese betrifft die zugrundeliegende Studienarbeit jedoch nicht und ist nur zur Vollständigkeit aufgeführt.

2 Aufgabenstellung

Die Aufgabenstellung ist es die Bewegung eines springenden Balles in MATLAB zu simulieren und anschließend zu visualisieren.
Die Analyse physikalischer Zusammenhänge einer beschleunigten Bewegung soll dazu genutzt werden um eine Abbildung der realen Bewegung eines Balles im freien Fall in MATLAB darzustellen. Die Bewegung besteht aus einer Phase im freien Fall, der Aufprallphase und der Phase des Rückprallens.



3 Prozessdynamik

Bei dieser Anwendung soll die Gravitationskraft der Erde visualisiert werden. Somit handelt es sich um eine beschleunigte Bewegung.

Andere Kräfte wie die Luftreibung (Stokes- und Newton-Reibung) werden in dieser Implementierung aufgrund des beschränkten Zeitfensters der Bearbeitung vernachlässigt.

Somit lässt sich die Bewegung anhand folgender Formeln beschreiben:

$$v(t) = \int a(t)\,dt + v_o$$

$$s(t) = \iint a(t)\ dt^2 + v_0 * t + s_0$$

Hierbei ist die Beschleunigung mit

a = 9,81 N/kg

konstant damit gilt:

$$h(t) = \frac{1}{2} a t^2 + v_0 * t + h_0$$

Der Aufprallmoment wird durch h(t)<=0 detektiert und über eine Invertierung der Geschwindigkeit realisiert.

Die Reibung beim Aufprall wird über eine Multiplikation der Geschwindigkeit und einer Zahl zwischen Null und Eins simuliert.

3.1 Implementierung der Prozessdynamik

Wie zuvor gezeigt wurde, kann die Position des Balls anhand der Parabelgleichung, wie auch des Integrals beschrieben werden.

Für die Applikation in MATLAB wurde eine approximierte Lösung des Integrals gewählt.
Diese nutzt einen festgelegten Zeitschritt, um so die Änderungen wie Δv und somit auch Δh zu berechnen.

Dies bietet den großen Vorteil, die Flugbahn einfach durch äußere Einflüsse zu beeinflussen.
Dazu kann einfach die momentane Geschwindigkeit über Einlesen der Pfeiltasten manipuliert werden.

Hier der dazu implementierte Code:

```
%% Berechnung der neuen Position
tic;toc;
pos = pos + v*dt(1);
v = v + a * dt(1);
```

Die Befehle *tic* und *toc* werden dazu verwendet, die Zeit eines Aktualisierungsprozesses zu erfassen.

Wenn *dt(1)* gleich wie diese Zeit gewählt wird, kommt eine realistische Bewegung zustande. Ansonsten ist sie verzerrt.

```
%% Manipulation der Geschwindigkeit
if strcmp(input,'leftarrow')
    v(1) = v(1)-1;
    input = 'return';
end
if strcmp(input,'rightarrow')
    v(1) = v(1)+1;
    input = 'return';
end
if strcmp(input,'uparrow')
    v(2) = v(2)+1;
    input = 'return';
end
if strcmp(input,'downarrow')
    v(2) = v(2)-1;
    input = 'return';
end
```

4 Prozessvisualisierung

Für die Visualisierung des Prozesses wurde eine Vektorgrafik verwendet. In diese wurden verschiedene Objekte eingebunden.

Eine horizontale Linie bildet den Boden, welche von zwei weiteren vertikalen Linien (den Wänden) begrenzt.

Da kein Kreisobjekt für die Nachbildung des Balles vorhanden ist, wird ein Rechteck mit abgerundeten Ecken genutzt.

Die Position des Rechtecks wird mit bei jeder Aktualisierung angepasst.

Die Bewegung des Balls wird somit visualisiert. Der Ball als Objekt bleibt dadurch immer erhalten; es werden nur die entsprechenden Eigenschaften aktualisiert.

Als Beispiel ist an dieser Stelle ein Listing zu sehen, wie das Rechteck initialisiert und aktualisiert:

```
kreis = rectangle('Curvature',[1 1],'FaceColor',farbe_kugel);
```

Initialisierung des Kreises:
das Attribut *Curvature* beschreibt
die Rundung der Ecken
→ Rechteck wird zu Kreis

```
%% Aktualisierung der Kreisposition
kreis.Position = [pos-r_ball/2 r_ball r_ball];
```

Um die Position zu aktualisieren wird der Eigenschaft *Position* des Kreises ein neuer Wert zugewiesen.

5 Ausführung und Bedienung

Das Ergebnis ist eine in MATLAB implementierte Anwendung, die die Bewegung eines springenden Balles anhand einer Vektorgrafik simuliert.

Die Bewegung kann durch die Pfeiltasten beeinflusst werden.

Somit wurden alle gestellten Anforderungen erfüllt und sogar übertroffen.

Zusätzlich werden Prozesswerte mittels Textobjekten im oberen Bildschirmbereich dargestellt.

Hierzu gehört die aktuelle *x* und *y* Position und der Betrag der Geschwindigkeitsvektoren derselben Richtung.

Die Summe der potentiellen und kinetischen Energie wird ebenfalls angezeigt.