Modellierung dynamischer Systeme Entwurf zur Bearbeitung der Praktikumsaufgabe 3

Maria Lüdemann und Birger Kamp

May 11, 2016

1 Schiefer Flipper

Diese Praktikumsaufgabe beschäftigt sich mit dem simulieren eines 'Flippers'. Dafür soll die Bewegung einer Kugel auf einer geneigten Ebene simuliert werden. Die Ebene ist von drei Wänden begrenzt und hat in der Mitte ein zylindrisches Hindernis. Das Abprallen des Balles von den Wänden und dem Hindernis verändert die Geschwindigkeit des Balles sowie die Richtung seiner Bewegung.

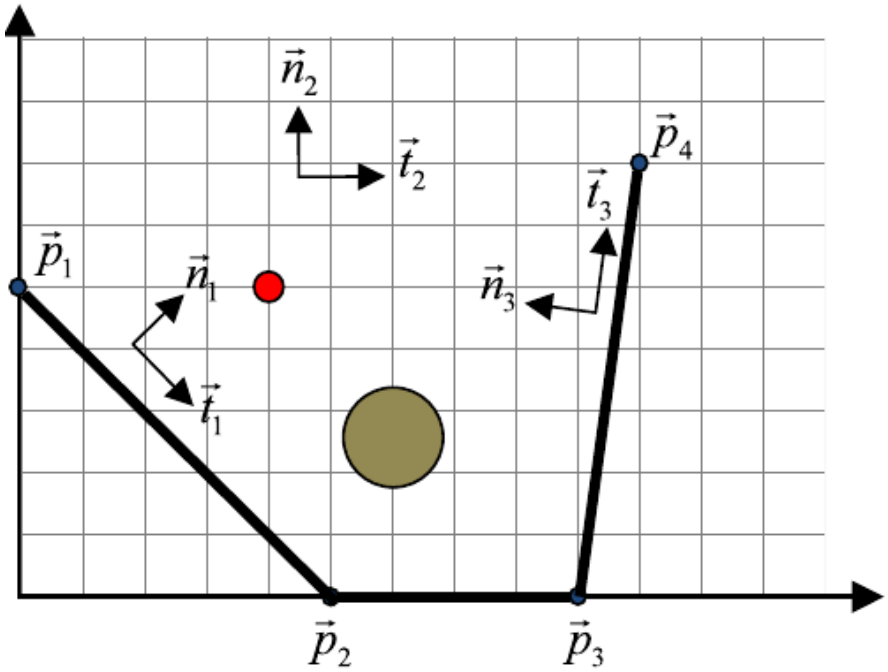

Figure 1: Skizze zum schiefen Flipper

1.1 Gegebene Formeln und Konstanten

Die Begrenzenden Wänder, durch Vektoren dargestellt definieren sich als:

$$\vec{p_1} = (0,5)^T \ \vec{p_2} = (5,0)^T \ \vec{p_3} = (9,0)^T \ \vec{p_4} = (10,7)^T \tag{1}$$

Die Position und der Umfang des zylindrischen Hindernisses entsprechen:

$$\vec{p_Z} = (6,2.5)^T \ RHnd = 0.8 \tag{2}$$

Startposition der Kugel

$$\vec{xa}_0 = (4,5)^T \tag{3}$$

Startgeschwindigkeit der Kugel

$$\vec{v1}_0 = (0,0)^T \tag{4}$$

Radius der Kugel

$$R = 0.25 \tag{5}$$

1.2 Simulationsrandbedingungen

Die Variablen Konstanten und Parameter für diese Aufgabe sind stark vorgegeben und werden dem der Aufgabenstellung beigefügten Bild entnommen. Dieses Bild ist untenstehend gezeigt. Dabei handelt es sich bei $RHnd$ um den oben definierten Radius des Zylinderhindernisses. R ist der oben definierte Radius der Kugel und Hnd der Ort, also Positionsvektor des Zylinderhindernisses wie oben als $\vec{p_Z}$ bestimmt.

1.3 Funktionen

Um das System zu simulieren wurden einige Funktionen vorgegeben die entworfen werden müssen.

1.3.1 Init

Die Funkion Init die die folgenden benötigten Größen initialisiert:

- Wandeckpunkte $(\vec{p_1}, \vec{p_2}, \vec{p_3}, \vec{p_4})$und Hindernispositionen $\vec{p_Z}$
- Zustandsgrößen
- die Wand-Tangential und Normalvektoren,
- Startposition $\vec{xa}_0$ und Startgeschwindigkeit $\vec{v1}_0$ der Kugel

1.3.2 Acceleration

Die Funktion $Acc()$ soll die Beschleunigung a der Kugel berechnen.

1.3.3 Wand..Kontakt

Für jede der drei Wände muss es eine Funktion geben die den Kontakt zwischen der Wand und der Kugel feststellt. Dafür wird ein *true* zurück gegeben wenn die Kugel in der Vorwärtsbewegung die Wand berührt.

1.3.4 Wand..Reflexion

Für jede der drei Wände muss eine Funktion geschrieben werden die wenn die dazugehörige Wand..Kontakt Funktion *true* liefert die Reflexion der Kugel von der Wand realisiert.

1.3.5 Hindernis Kontakt

Es wird eine Funktion benötigt die den Kontakt mit dem Hindernis berechnet und genau wie $Wand..Kontakt$ ein *true* zurück gibt wenn die Kugel das Hindernis in der Vorwärtsbewegung berührt.

1.3.6 Hindernis Reflexion

Um die Reflexion der Kugel am Hindernis zu simulieren muss eine Funktion $HndRefl()$ geschrieben werden die die Reflexion der Kugel berechnet wenn die dazugehörige $HndKontakt$ ein $true$ liefert.

1.3.7 Zustand

Der Zustand selber soll in der *during*-Section die Differentialgleichungen und die Übergabe an die Ausgangsgröße abhandeln.

1.3.8 Simulation

Die Simulation soll mit dem Programm $Table.wrl$ dargestellt und visualisiert werden.

1.4 Versuchsdurchführung

Die Simulation soll mit den Einstellungen $Non-adaptive$ für den $Zero-crossingalgorithm$ gestartet werden. Dafür gibt es eine vorgegeben Einstellung in der Aufgabenstellung für die VR-Sink. Das folgende Bild ist aus der Aufgabenstellung übernommen und zeigt die nötigen Einstellungen.

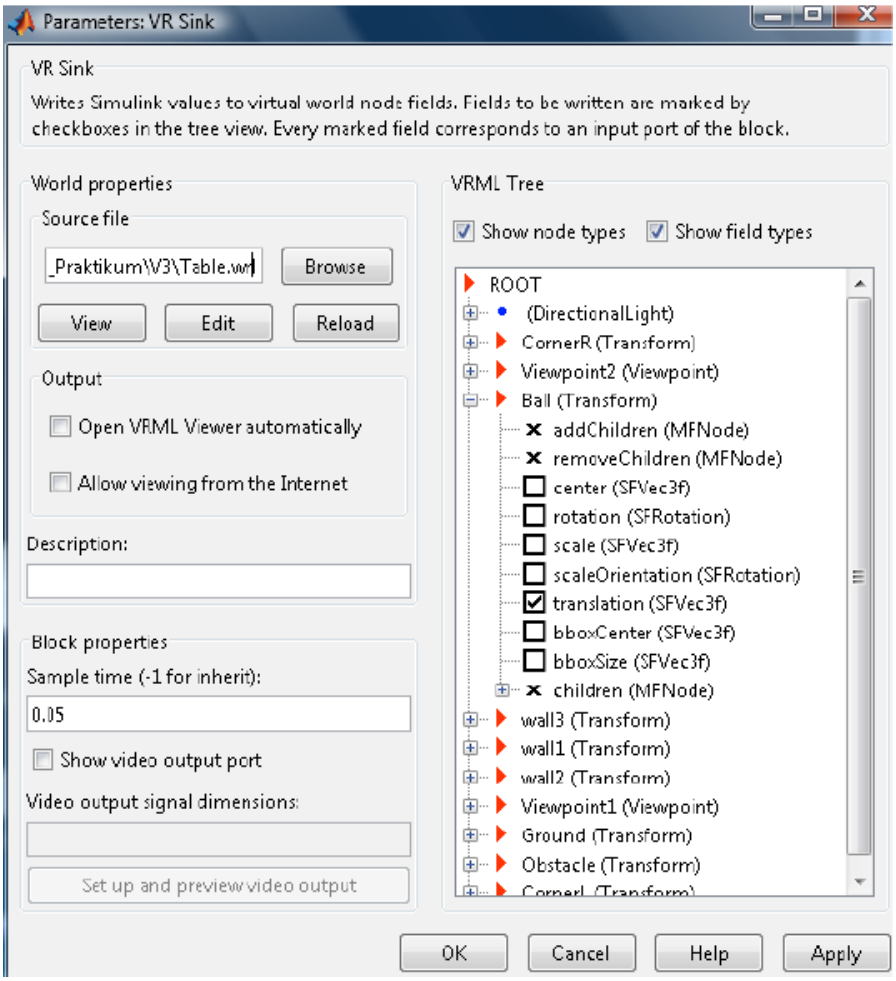

Figure 2: Einstellung für die Versuchsdurchführung