UNIVERSITÄT
LEIPZIG

Softwaretechnikpraktikum

Rechercheberichtt

Gruppe:	nw19a
Mitglieder:	Thomas Pause, Sabine Lorius, Arik Korte, Martin George, Josephine Lange, Esther Prause, Anh Kiet Nguyen, Bärbel Hanle
Verantwortlich:	Esther Prause
Betreuer:	Dr. Nicolas Wieseke
Tutor:	Martin Frühauf
Abgabedatum:	11.11.2019

Stand: 11. November 2019

Inhaltsverzeichnis

1 Aspekte

1.1 Ausgangssituation

Der Auftraggeber möchte einen mathematisch-physikalisch fundierten Machbarkeitsbeweis für die Simulation von Poi-Figuren in Form einer Android-Anwendung. Es gibt bereits einige wenige Programme zur Simulation von Poi-Figuren, z.B. Poi Lab[1], VTG[2] und Poi Simulator[3], diese arbeiten jedoch ohne Berücksichtigung jeglicher physikalischer Gegebenheiten auf festen Bahnen.

1.2 Zielvorstellung

Ziel ist die Entwicklung einer nativen Android-App als Proof-Of-Concept, die es Poi-Spielern ermöglicht zu berechnen, auf welche Weise (beschränkt auf den 2-dimensionalen Raum) beliebige Poi-Figuren erzeugt werden können. Als einfachste Stufe sollen mit beliebigen Eingabeparametern sog. Flowers berechnet werden. Als weitere Funktionalität soll sowohl aus einer Touch-Eingabe als auch aus den Außenkonturen eines Eingabebildes eine parametrisierte Kurve erstellt werden können, die die gezeichnete Bahn eines Poi-Kopfes darstellt. Auf dieser Grundlage soll eine neue parametrisierte Kurve errechnet und grafisch dargestellt werden, die die zur Erzeugung der Eingabekurve theoretisch nötige Handbewegung angibt. Den mathematischen Ansatz liefert hier die komplexe Fourieranalyse. Schließlich soll eine Simulation der Poi-Bahn unter Eingabe der berechneten Handbewegungen und unter zusätzlicher Berücksichtigung physikalischer Aspekte erfolgen.

1.3 Gestaltungsspielräume

Gelingt eine Implementierung von komplexeren Bilderkennungsverfahren und Algorithmen zur Kontur-Erkennung, wäre das Erzeugen parametrisierter Kurven nicht nur aus simplen Outlines von Objekten in Bildern möglich, sondern auch aus sich überschneidenden oder ineinander verschachtelten Linienzügen auf einer durch die App eingelesenen Grafik.
Da unter Berücksichtigung physikalischer Bedingungen erhebliche Abweichungen zwischen Eingabekurve und simulierter Poi-Bahn zu erwarten sind, könnte die Abweichung zwischen diesen Kurven berechnet werden und ein korrigierter Vorschlag für Handbewegungen erfolgen. Dies könnte ggf. wiederholt werden, um die Abweichung schrittweise zu verringern.
Die ermittelten Handbewegungen könnten durch eine Animation der Gliedmaßen ergänzt werden.

1.4 Frontend

Das Frontend muss das Poi-Fachvokabular berücksichtigen und ist in drei wichtige Aspekte gegliedert: Menüführung, Userinput und Ausgabe/Visualisierung.
Für den Endanwender wird eine übersichtliche Menüführung benötigt. Die Eingabe beliebiger Parameter zur Berechnung der Flowers ist durch ein entsprechendes Interface bereitzustellen. Außerdem soll dem Nutzer die Möglichkeit gegeben werden, beliebige Muster direkt über die Touch-Eingabe seines mobilen Endgerätes einzugeben. Möglicherweise bietet es sich hier an, den zeitlichen Verlauf des Zeichnens direkt aufzunehmen und für die Parametrisierung der Kurve zu nutzen. Schließlich sollen die Handbewegungen und Laufbahnen der Pois ansprechend gestaltet auf dem Display ausgegeben werden.

1.5 Algorithmischer Teil

Der Nutzer soll spielerisch erkunden können, welchen Einfluss Parameter wie das Verhältnis zwischen Radien und Frequenzen der Kreisbewegungen, aus denen Flowers zusammengesetzt sind, auf die Figur haben. Neben der Möglichkeit, eine Kurve durch Eingabe der Parameter und über das Malen von

Linienzügen auf dem Touchscreen zu erzeugen, soll die Anwendung aus Bildern die für die Berechnung nötigen Punkte extrahieren können. Dafür bedarf es einer Reihe von Image-Processing-Verfahren, um ein vom User bereitgestelltes Bild in der Art vorzubereiten, dass das zu zeichnende Objekt klar vom Hintergrund abgegrenzt werden kann. Mit Kontur-Algorithmen kann dann eine Outline im Bild erkannt werden, aus der wiederum eine Kurve generiert werden kann.
Zu einer vorgegebenen Handbewegung, die in Form einer parametrisierten Kurve vorliegt, soll eine Simulation erfolgen, wie sich der Poikopf unter Beachtung physikalischer Gesetze bewegen würde. Dies könnte durch Einbinden einer Physik-Engine geschehen. Da die numerische Approximation der Poibewegung jedoch nicht schwierig ist, wenn man den Poikopf als Punktteilchen und die Schnur als masselos annimmt, kann die Poibahn auch in eigener Ausführung berechnet werden.

2 Konzepte

2.1 SCRUM

SCRUM, 2005 von Schwaber und Sutherland entwickelt, ist ein Rahmenwerk für eine agile Projektmanagementmethode, das auf kontinuierliche Verbesserung im Team (Lean) sowie eine durch empirische Prozesssteuerung geprägte, messbare Vorgehensweise setzt. Verschiedene Techniken, Prozesse und Dokumentationen werden mittels der SCRUM-Regeln und verteilten Rollen, Ereignisse und Artefakte organisiert und erstellt. Sie ermöglichen hierbei, das zu entwickelnde Softwareprodukt, das Team sowie die Arbeitsumgebung sukzessive zu optimieren. Durch den iterativen, inkrementellen Ansatz von SCRUM (Sprints, Reviews, etc.) können Risiken und Fehler frühzeitiger erkannt, besser abgeschätzt und rechtzeitig behandelt werden. Vgl. [4].

2.2 git

Git[5] ist eine leicht zu erlernende freie Software zur verteilten Versionsverwaltung von Sourcecode und anderen Dateien. Sie wurde von Linus Torvalds initiiert und ist inzwischen der quasi-Standard der Versionskontrolle, vor allem aufgrund der Vielfältigkeit der Funktionen und der Performance.

2.3 GitLab

GitLab[6] ist eine auf git aufgesetzte webbasierte Anwendung, die es ermöglicht, komfortabel auch große Softwareprojekte zu verwalten. GitLab bietet viele Management- und Bug-Tracking-Funktionalitäten sowie ein System zur kontinuierlichen Integration (GitLab-CI). U.a. kann man mit einem Issue-Board arbeiten, welches in übersichtlicher Form die Abarbeitung von Teilaufgaben visualisiert. Weiterhin lässt sich ein Projekt in Teilprojekte (sog. Milestones) unterteilen, denen man jeweils ein Ablaufdatum und eine Menge von zu erledigenden Teilaufgaben zuordnen kann.

2.4 Pairprogramming

Beim Pairprogramming gibt es zwei Rollen: Den Piloten (Driver) und den Navigator (Partner). Der Pilot bedient den Rechner und schreibt den Code. Der Navigator kontrolliert die Korrektheit des Codes, gibt Feedback und denkt über Verbesserungen nach. Somit kann sich der Pilot auf die Details des Codeabschnitts konzentrieren, während der Navigator den Ablauf überwacht. Die Rollen werden regelmäßig getauscht, auch die Paarzusammensetzung sollte häufig rotieren. Vgl. [7]

2.5 Continuous Integration

CI[8] beschreibt den Prozess des kontinuierlichen Zusammenfügens von Komponenten zu einer Anwendung. Sie ist ein Werkzeug zur Steigerung der Softwarequalität. Hierzu werden bei neu geschriebenen Modulen/Teilen der Anwendung automatisierte Tests (z.B. Build-Tests) oder Metrikprüfungen durchgeführt. Bei der GitLab-CI geschieht dies beim Push der Änderungen.

2.6 Test-driven development

TDD[9] ist eine Designstrategie in der Softwareentwicklung, in der Tests noch vor zu implementierenden Funktionalitäten geschrieben werden. Vorgehen (durchläuft den sog. 'red-green-refactoring'-Zyklus):
- Schreibe einen Unit-Test für eine einzelne Komponente. Der erste Testlauf wird scheitern, da der erforderliche Code noch nicht vorhanden ist ("red").
- Schreibe Code, bis der Test erfolgreich ist ("green"). In diesem Schritt soll nur auf *diesen* Test zugeschnittener Produktivcode entstehen.
- Refactoring von Test und Code zur Qualitätsverbesserung ("refactoring").
Dieser Zyklus wird für alle weiteren noch fehlenden Komponenten wiederholt.

2.7 LaTeX

LaTeX ist ein open-source betriebssystemunabhängiges Textsatzsystem, das in den 1980er Jahren von Leslie Lamport als Makropaket basierend auf TeX-Basisbefehlen (entwickelt von Prof. Dr. Knuth in Stanford) geschnürt und seither um hunderte dazuladbare Makropakete ergänzt wurde. Aktuell wird Version 2ε genutzt. Neben dem perfekten Satz mathematischer Formeln bietet es die Möglichkeit Bilder, Tabellen und bspw. Notensatz einzubinden. Auf Grundlage der SGML (Structured Generalized Markup Language) trennt LaTeX die Struktur eines Textes von dessen Inhalt. Vgl. [10].

2.8 Android

Android ist sowohl ein Betriebssystem als auch eine Software-Plattform für mobile Endgeräte wie Smartphones, Fernseher, Tablets etc., die von der von Google gegründeten Open Handset Alliance entwickelt werden. Es ist an keine spezifische Hardware gebunden und viele Android-Applikationen werden in Java geschrieben. Vgl. [11]

2.9 AndroidStudio

AndroidStudio[12] ist eine auf IntelliJ basierende IDE von Jetbrains (Google), die es ermöglicht, auch interaktiv Android-Apps zu erstellen. Dies geschieht über die Trennung von Layout/Design und funktionalem Code. Als Buildtool wird Gradle genutzt. Für Inline- und technische Dokumente lässt sich JavaDoc verwenden. Seit 2017 wird neben Java auch Kotlin unterstützt. Um Android Studio vollumfänglich nutzen zu können, muss BIOS-seitig die Virtualisierung aktiviert sein. Vgl. [13].

2.10 Java Android SDK

Java Android SDK[14] ist ein Software-Development-Kit für Android OS, i. e. eine Menge von Entwicklungsanwendungen für die Android-Plattform, die auf dem Java SDK basiert und auf die speziellen Bedürfnisse des mobilen Betriebssystems angepasst ist.

2.11 Android Scripting Environment (ASE)

Mit dem Android Scripting Environment (ASE)[15] von Google können Entwickler im Android-Umfeld Skripte in Python, Ruby, Lua und BeanShell auf dem Gerät schreiben und ausführen. Diese haben Zugriff auf viele der Android-APIs und lassen sich zum Beispiel einsetzen, um Aktivitäten zu initiieren, SMS-Nachrichten zu verschicken, Telefonate zu starten, Text in Sprache umzusetzen (TTS) sowie um den aktuellen Standort und andere Sensor-Informationen auszulesen.

2.12 Android Native Development Kit (NDK)

Das Android NDK[16] ist eine Menge von Entwicklungsanwendungen für C(++)-Komponenten.

2.13 Jekyll

Jekyll[17] ist ein in Ruby geschriebener Static Website Generator. Er baut statische HTML-Seiten mithilfe von MarkDown-Dateien. Dies kombiniert die Vorteile Lesbarkeit und Erweiterbarkeit (HTML-Code usw.). Da Jekyll einen eigenen Server mitbringt, kann man live Änderungen verfolgen, die auf der Basis von Design-Vorlagen (sog. Themes) vorgenommen werden. Das Konzept eignet sich bspw. für Blogs und kleinere Webseiten. Im Rahmen des Praktikums wird das Hyde Theme verwendet, welches ein einfaches 2-spaltiges Design mit Navigation mitbringt und sehr gut auf unterschiedlichen Geräten (PC, SmartTV, Tablet, Smartphone) skaliert. Vgl. auch [18].

2.14 OpenCV

Die Open Source Computer Vision Library ist eine Computer-Vision-Bibliothek, die in C++, Java oder Python implementiert werden kann. Sie stellt eine Vielzahl an Algorithmen und Verfahren aus den Bereichen Image-Processing, Image-Recognition, Machine-Learning und anderen zu Verfügung. Des Weiteren kommt OpenCV sowohl in vielen kommerziellen Produkten als auch in Open-Source-Projekten zum Einsatz, was Benutzern eine sehr aktive Community als Ressource zur Seite stellt. Mithilfe der in der Bibliothek enthaltenen Verfahren kann man Bilder für die Extraktion von Kurven vorbereiten sowie Kurven anhand von Konturen generieren, um sie dann der Berechnung zuzuführen. Vgl. [19].

2.15 Ionic

Ionic[20] ist ein Framework zum Erstellen hybrider Apps sowie von Progressive Web Apps. Es basiert auf HTML5, CSS, Sass sowie Javascript/Typescript (einer typisierten Obermenge von Javascript). Des Weiteren bilden Angular und Apache Cordova die Basisframeworks, jedoch kann ionic auch auf anderen Frameworks wie z.B. React oder Vue.js aufgesetzt werden.

2.16 Touch-Interaktion zur Kurveneingabe

Für die Touchpadeingabe einer 2-dimensionalen Figur auf dem Smartphone/Tablet stellt die Android SDK alle nötigen Mittel bereit. Grob besteht eine Zeichenanwendung aus 3 Schritten: 1. Erstellung eines Path-Objektes, das via android.graphics.Path bereitgestellt wird. 2. Erstellung einer Custom-View (extends View), auf der ein Canvas mit einer Bitmap erstellt wird. 3. Managen des Userinputs via Überschreiben der onTouchEvent()-Methode. Vgl. [21].
Mit den Klassen PathMeasure und RecordingCanvas ist evtl. Parametrisierung möglich. Vgl. [22, 23].

2.17 Kotlin

Kotlin[24] ist eine von JetBrains entwickelte Programmiersprache, die neben Java in Android Studio für die Android App-Entwicklung genutzt werden kann. Kotlin ist syntaktisch nicht mit Java kompatibel, Java-Code kann jedoch in Kotlin inkorporiert werden. Kotlin hat gegenüber Java einige Vorteile, etwa eine einfachere Syntax und das Unterstützen von Typinferenzen.

2.18 Frameworks/Libraries für mathematisch-physikalische Berechnungen

FFTW[25]	Java-Library, die unter anderem Tools zur Berechnung von diskreten Fourier-Transformationen (max. dritten Grades) zu Verfügung stellt.
ND4J[26]	Java-Library, die das Benutzen n-dimensionaler Arrays in Java ermöglicht und einige Lineare-Algebra-Tools bereitstellt.
NumPy[27]	Der Standard unter den Python-Packages für Berechnungen mit großen n-dimensionalen Arrays. Stellt eine Vielzahl an Funktionen für numerische Berechnungen, Matrixoperationen etc. bereit.
SymPy[28]	SymPy kann als Python-Package oder eigenständige Software benutzt werden. Neben einer Reihe an Funktionen für Grundrechenarten können mit SymPy auch effizient komplexere Berechnungen wie das Lösen von Differentialgleichungen realisiert werden. Anders als NumPy bietet SymPy zudem eine Schnittstelle für die grafische Darstellung von Berechnungen.
networkx[29]	Python-Package für das Arbeiten mit Graphen. Erlaubt es, Datenstrukturen als Graphen zu speichern und alle gängigen Operationen auf Graphen auf effiziente Weise auszuführen.
matplotlib[30]	Python-Package für mathematische Visualisierung. Mit Matplotlib können relativ niederschwellig Kurven, Graphen, Diagramme etc. erzeugt und optisch ansprechend dargestellt werden.

2.19 Algorithmen zur Linienfindung

Um Konturen aus einem Bild zu extrahieren, benutzt die OpenCV-Bibliothek den Border-Following-Algoritmus von Suzuki und Abe [31]. Dieser findet relativ zuverlässig Außenkonturen, ignoriert jedoch innen liegende Konturen. Über eine Modifikation oder eine erneute Anwendung auf einen Teilausschnitt des Bildes könnten auch innere, in sich geschlossene Linien gefunden werden. Dafür könnten auch Edge-Detection-Verfahren[32] zur Anwendung kommen (etwa Canny). Eine weitere Problemstellung ist es die Linienführung zu erkennen, wenn sich Segmente überschneiden. Eine mögliche Lösung dafür liefert Raghupathy[33]. Werden mehrere Konturen aus einem Bild extrahiert oder bestehen Konturen aus nicht geordneten Punkten (was bei Border-Following nicht der Fall ist), stellt sich die Frage der Reihenfolge der zu besuchenden Punkte. Dieses Problem ähnelt im Allgemeinen dem TSP-Problem (Traveling Salesperson), für das es eine Reihe an Lösungsansätzen[34] gibt. Können für diese Problemstellungen Lösungen gefunden werden, so könnte in der Anwendung auch die Möglichkeit implementiert werden, parametrisierte Kurven aus Bildern zu gewinnen und dabei jede Kontur zu beachten.

2.20 Numerisches Lösen von Differenzialgleichungen

Die Bewegung des Poikopfs unterliegt der Newtonschen Bewegungsgleichung. Um diese Bewegung für die Simulation der Poibahn zu berechnen, muss also eine Differenzialgleichung gelöst werden. Es existiert eine Reihe von numerischen Verfahren, mit denen sich dieses Problem näherungsweise lösen lässt. Das einfachste ist das sogenannte Tangentenverfahren (auch Eulerverfahren genannt), mit dem

man, wenn der Ort $\vec{x}(t)$ und die Geschwindigkeit $\vec{v}(t)$ des Poikopfs zu einem Zeitpunkt t bekannt sind, diese Größen für den nächsten Zeitpunkt $t + \Delta t$ berechnen kann. Dabei ist Δt ein kleines Zeitintervall. Vgl. [35, 36].

2.21 Physik-Engines

Box2D[37] und LiquidFun[38] sind zwei für die Android-Nutzung geeignete Physik-Engines. Box2D ist eine in C++ geschriebene, kostenlose Open-Source 2D-Physik-Engine. Sie wird in Spielekonsolen wie Nintendo Wii oder Nintendo DS als auch in Betriebssystemen wie Android und iOS verwendet und kann auf jeder Plattform mit einem C++-Compiler eingebunden werden. LiquidFun ist eine 2D-C++-Bibliothek zur Simulation von Starrkörpern und Flüssigkeiten für Spiele und Animationen. Es bietet Unterstützung für die prozedurale Animation physischer Körper, um Objekte in realistischer Weise zu bewegen und interagieren zu lassen. LiquidFun-Code kann in C++, Java oder JavaScript geschrieben werden.

2.22 Linksammlung

Übersicht Android Grafik-APIs: https://developer.android.com/guide/topics/graphics/
Einführung zu Responsive Design in Android: https://medium.com/androiddevelopers/building-a-responsive-ui-in-android-7dc7e4efcbb3
DFT-, FFT-Implementierungen mit Python: http://jakevdp.github.io/blog/2013/08/28/understanding-the-fft/
NumPy fft-Dokumentation: https://docs.scipy.org/doc/numpy/reference/routines.fft.html

3 Begriffe

3.1 Computergrafik (Themengebiet)

Bitmap[39]	Auch Rastergrafik. Bildbeschreibung in Form von computerlesbaren Daten. Das Bild besteht aus Pixeln (Bildpunkten), denen eine Farbe zugeordnet ist. Bildgröße = Bildhöhe und -breite in Pixeln. Dateigröße proportional zur Auflösung/Qualität. Besonders geeignet für Darstellung komplexerer Bilder. Dateiformate u.a.: JGP (Kompression mit deutlicher Reduktion der Dateigröße, geeignet für Fotos), GIF (Animationen möglich, Farbtabelle max. 256 Farben/8 Bit), PNG (kann Interlacing, keine Animationen; verlustfreie Kompression bei schwacher Reduzierung der Dateigröße).
Interlacing[40]	Speicherverfahren für Rastergrafiken, das einen beschleunigten Bildaufbau beim Laden ermöglicht.
Vektorgrafik[41]	Aus graphischen Primitiven zusammengesetzte Computergrafik. Speichersparend, stufenlose und verlustfreie Transformationen. Eigenschaften von Objekten lassen sich beliebig anpassen. Wiedergabe auf Bildschirmen erfordert Rasterung (Kosten!).
SVG[42]	Scalable Vector Graphics. Auf XML basierende Spezifikation zur Umschreibung von 2D-Vektorgrafiken. Kann responsives Design, verlustfrei skalierbar. Animationen möglich.
Vektorfonts[43]	V. umschreiben Umrisse. Outline-Fonts beschreiben die Kontur mathematisch, z.B. Open Type, True Type, PostScript. Stroke-Fonts definieren Striche und ihre Stärken.
Vektorisierung[44]	Auch (Bild-)Tracing. Automatisierte Umwandlung einer Raster- in eine Vektorgrafik.

Rendern[45]	Auch Rasterung, Bildsynthese. Erzeugung eines Bildes aus Rohdaten. In 2D-Computergrafik: Umwandlung einer Vektor- in eine Rastergrafik.
Skalierung[46]	Größenänderung eines digitalen Bildes. S. von Rastergrafiken ändert die Bildauflösung (oft Qualitätsverlust). Verlustfreie S. bei Vektorgrafiken.
Responsive UI[47]	User Interface, das auf geräteabhängigen verfügbaren Bildschirmplatz reagiert.

3.2 Fourieranalyse

Komplexe Fourieranalyse ermöglicht es, eine parametrisierte Kurve in der Gaußschen Zahlenebene als eine Überlagerung von Kreisbewegungen mit ganzzahligen Frequenzen darzustellen. Vgl. [48].

3.3 Gaußsche Zahlenebene

Die Gaußsche Zahlenebene[49], auch komplexe Ebene, ist eine geometrische Repräsentation der komplexen Zahlen. Sie stellt jede komplexe Zahl als Punkt in dieser Ebene dar, wobei die x-Achse die Teilmenge der reellen Zahlen (also den Realteil) und die y-Achse die Teilmenge der rein imaginären Zahlen (also den Imaginärteil) repräsentiert. Eine komplexe Zahl $z = a + bi$ ist also der Punkt mit dem (x, y)-Tupel (a, b).

3.4 Newtonsche Bewegungsgleichung

Nach dem zweiten Newtonschen Gesetz ist die Änderung der Geschwindigkeit eines Körpers proportional zu der Kraft, die diese Veränderung hervorruft. Dies führt für Punktteilchen mit konstanter Masse m zu folgender Differenzialgleichung:

$$\vec{F}(t) = m\frac{d\vec{v}(t)}{dt}$$

Diese Gleichung erlaubt es die Bahn eines Teilchens für alle Zeiten zu berechnen, wenn zu einem Zeitpunkt Ort und Geschwindigkeit (Anfangsbedingungen) bekannt sind. Vgl. [35].

3.5 Poi (Themengebiet)

Ein Poi besteht konzeptuell aus einer Kugel, an der sich eine Schnur mit einem Griff oder einer Schlaufe am anderen Ende befindet, von wo aus die Kugel im Kreis geschwungen werden kann. Durch zusätzliche Bewegungen der Hand kann eine Vielzahl an unterschiedlichen kreisähnlichen Bahnen erzeugt werden. Vgl. [50]
Flowers: Wird der Arm, welcher den Poi hält, zusätzlich in Kreisbahnen bewegt, entstehen in Abhängigkeit von Rotationsrichtung und Geschwindigkeit unterschiedliche Muster. Da viele dieser Figuren schlaufenförmige sog. 'Petals' aufweisen, werden sie Flowers genannt. Vgl. [50]

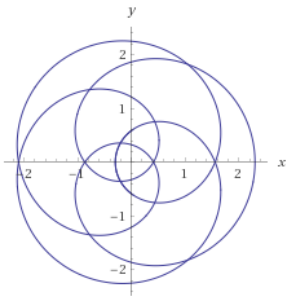
Poi-Terminologie: (eine Auswahl)

Beats	Anzahl von Poi-Rotationen
Wall-, Wheel-, Horizontal-Plane	Ausrichtung der Poi-Ebenen zum Spieler
Parallel / Mirror	Rotationsrichtung der Poi zueinander
Same Time, Split Time, Quarter Time...	Offset der Poi-Rotationen
In-Spin, Pro-Spin, Anti-Spin, Isolation, Hybrid...	Rotation der Arme relativ zu den Poi-Rotationen
Stalls	Anhalten/Richtungswechsel der Poi

Vgl. [51].

3.6 Parametrisierte Kurven

Eine parametrisierte Kurve in $\mathbb{R}^n$ kann durch eine Funktion $\delta : M \subset \mathbb{R} \to \mathbb{R}^n$ dargestellt werden, wobei M ein Intervall ist. Somit kann $t \to \delta(t)$ als Bewegung eines Punktes in $\mathbb{R}^n$ verstanden werden, wodurch von $\delta(M)$ eine durchlaufende Linie in $\mathbb{R}^n$ gezeichnet wird. Beispielsweise lassen sich Flowers durch parametrisierte Kurven der Form



$$x(t) = cos(2\pi t) + r * cos(2\pi\omega t + \varphi), \quad y(t) = sin(2\pi t) + r * sin(2\pi\omega t + \varphi)$$

darstellen, wobei r, ω und φ frei wählbare Parameter sind. r entspricht dabei dem Verhältnis der Radien, ω dem Verhältnis der Frequenzen und φ der Phasenverschiebung von Hand- und Poikreis. Vgl. [52].

3.7 Physik-Engine

Eine Physik-Engine ist für die Simulation der Bewegungen und Reaktionen von Objekten verantwortlich, als wären sie den realen Bedingungen der Physik unterworfen. Die Eingabe für eine derartige Physiksimulation ist im Allgemeinen eine Sammlung von Objekten ("Körpern") mit Eigenschaften, sowie die Angabe der Kräfte, die auf diese Objekte wirken sollen. Vgl. [53]

3.8 Punktteilchen

Ein Punktteilchen ist die Abstraktion eines physikalischen Körpers unter Vernachlässigung seiner räumlichen Ausdehnung. Vgl. [35].

3.9 Schwerkraft

Da der Poikopf der Schwerkraft unterliegt, wollen wir diese in unserer Simulation berücksichtigen. Diese Kraft lässt sich durch die Gleichung

$$\vec{F}_g = mg * \vec{e}_{\downarrow}$$

berechnen, wobei m die Masse des Körpers ist, auf den die Kraft wirkt, g die Fallbeschleunigung ($g \approx 9.8\,\mathrm{m\,s^{-2}}$) und $\vec{e}_{\downarrow}$ ein Einheitsvektor, der nach unten zeigt. Vgl. [35].

3.10 Seildehnung

Dehnt man ein Seil, so entsteht eine Kraft, die das Seil auf seine ursprüngliche Länge zurückziehen will. Der Zusammenhang zwischen der ursprünglichen Seillänge l_0, der Seillänge im gedehnten Zustand l und dem Betrag der Kraft F_{Seil}, die der Dehnung entgegenwirkt, ist durch folgende Gleichung gegeben:

$$F_{Seil} = (l - l_0) * \alpha$$

wobei α ein vom Seil abhängiger Proportionalitätsfaktor ist. Vgl. [35].

Quellenverzeichnis

[1] *Android-App zur Simulation von Poi-Figuren (Poi-Lab).* URL: https://play.google.com/store/apps/details?id=net.firestaff.mcp.poilab.free (besucht am 09. 11. 2019).

[2] *Android-App zur Visualisierung von Poi-Theorie (VRG).* URL: https://play.google.com/store/apps/details?id=mcp.firestaff.net.VTGv3 (besucht am 09. 11. 2019).

[3] *Poi-Simulator von Danny Thomas.* URL: http://kaiein.com/poi_sim/sim6.htm (besucht am 09. 11. 2019).

[4] J. Böhm. *Erfolgsfaktor Agilität.* Wiesbaden: Springer, 2019.

[5] *Offizielle git-Dokumentation.* URL: https://git-scm.com/ (besucht am 18. 10. 2019).

[6] *Offizielle Webseite von GitLab.* URL: https://about.gitlab.com/ (besucht am 18. 10. 2019).

[7] Stephan Augsten chrissikraus. *Was ist Pair Programming?* URL: https://www.dev-insider.de/was-ist-pair-programming-a-788269/ (besucht am 09. 11. 2019).

[8] *Einführung in GitLab-CI.* URL: https://docs.gitlab.com/ee/ci/quick_start/ (besucht am 11. 11. 2019).

[9] *Wikipedia: Test-driven development.* URL: https://de.wikipedia.org/wiki/Testgetriebene_Entwicklung (besucht am 11. 11. 2019).

[10] Braune K. Lammarsch und M. Lammarsch. *LaTeX Basissystem, Layout, Formelsatz.* Heidelberg: Springer, 2006.

[11] *Wikipedia: Android.* URL: https://de.wikipedia.org/wiki/Android_(Betriebssystem) (besucht am 08. 11. 2019).

[12] *Offizielle Seite von Android Studio mit User Guide uvm.* URL: https://developer.android.com/studio (besucht am 11. 11. 2019).

[13] *Kleines Tutorial zum Umgang mit Android Studio.* URL: https://developer.android.com/training/basics/firstapp (besucht am 11. 11. 2019).

[14] *Wikipedia: Android SDK.* URL: https://de.wikipedia.org/wiki/Android-Softwareentwicklung#Android_SDK (besucht am 11. 11. 2019).

[15] *Google Blog zur Android Scripting Environment.* URL: https://opensource.googleblog.com/2009/06/introducing-android-scripting.html (besucht am 11. 11. 2019).

[16] *Einführung in Android NDK.* URL: https://developer.android.com/ndk (besucht am 11. 11. 2019).

[17] *Offizielle Webseite von Jekyll.* URL: https://jekyllrb.com/ (besucht am 18. 10. 2019).

[18] *GitHub-Repository des Hyde-Projektes.* URL: https://github.com/poole/hyde (besucht am 18. 10. 2019).

[19] *OpenCV Python Tutorials.* URL: https://docs.opencv.org/4.1.2/d6/d00/tutorial_py_root.html (besucht am 11. 08. 2019).

[20] *Offizelle Dokumentation des Ionic Framework.* URL: https://ionicframework.com/docs (besucht am 10. 11. 2019).

[21] Sylvain Saurel. *Learn to create a Paint Application for Android.* URL: https://medium.com/@ssaurel/learn-to-create-a-paint-application-for-android-5b16968063f8 (besucht am 09. 11. 2019).

[22] *Offizielle AndroidSDK-Referenz: PathMeasure Class.* URL: https://developer.android.com/reference/android/graphics/PathMeasure (besucht am 09. 11. 2019).

[23] *Offizielle AndroidSDK-Referenz: RecordingCanvas Class.* URL: https://developer.android.com/reference/android/graphics/RecordingCanvas (besucht am 09. 11. 2019).

[24] *Offizielle Referenz von Kotlin.* URL: https://kotlinlang.org/docs/reference/ (besucht am 09. 11. 2019).

[25] *Offizielles Manual von FFTW Version 3.3.8.* URL: http://www.fftw.org/fftw3.pdf (besucht am 11. 11. 2019).

[26] *Offizielle Seite von ND4J.* URL: https://nd4j.org/ (besucht am 11. 11. 2019).

[27] *Offizielle Dokumentation zu NumPy.* URL: https://numpy.org/devdocs/ (besucht am 11. 11. 2019).

[28] *Offizielle Dokumentation zu SymPy.* URL: https://docs.sympy.org/latest/index.html (besucht am 11. 11. 2019).

[29] *Offizielle Dokumentation von networkx.* URL: https://networkx.github.io/documentation/stable/ (besucht am 11. 11. 2019).

[30] *Offizielle Dokumentation von matplotlib.* URL: https://matplotlib.org/contents.html (besucht am 11. 11. 2019).

[31] Satoshi Suzuki und others. „Topological Structural Analysis of Digitized Binary Images by Border Following“. In: *Computer Vision, Graphics, and Image Processing* 30 (1985), S. 32–46.

[32] *Wikipedia: Edge-Detection.* URL: https://en.wikipedia.org/wiki/Edge_detection (besucht am 11. 08. 2019).

[33] Karthik Raghupathy. „Curve Tracing and Curve Detection in Images“. Magisterarb. Cornell University, 2004. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.571.3262&rep=rep1&type=pdf (besucht am 09. 11. 2019).

[34] *Wikipedia: Travelling Salesman Problem (TSP).* URL: https://en.wikipedia.org/wiki/Travelling_salesman_problem (besucht am 11. 08. 2019).

[35] *Physik des Poispiels.* URL: https://pcai042.informatik.uni-leipzig.de/~nw19a/Website/poiphysik/ (besucht am 10. 11. 2019).

[36] *Numerische Verfahren für Differentialgleichungen.* URL: https://www.informatik.uni-kiel.de/~sb/data/NumDgl.pdf (besucht am 09. 11. 2019).

[37] *Wikipedia: Box2D.* URL: https://en.wikipedia.org/wiki/Box2D (besucht am 08. 11. 2019).

[38] *Offizielle Webseite von LiquidFun.* URL: https://google.github.io/liquidfun/ (besucht am 08. 11. 2019).

[39] *Wikipedia: Rastergrafik.* URL: https://de.wikipedia.org/wiki/Rastergrafik (besucht am 11. 11. 2019).

[40] *Wikipedia: Interlacing (Grafiken).* URL: https://de.wikipedia.org/wiki/Interlacing_(Grafiken) (besucht am 11. 11. 2019).

[41] *Wikipedia: Vektorgrafik.* URL: https://de.wikipedia.org/wiki/Vektorgrafik (besucht am 11. 11. 2019).

[42] *Wikipedia: Scalable Vector Graphics.* URL: https://de.wikipedia.org/wiki/Scalable_Vector_Graphics (besucht am 11. 11. 2019).

[43] *Wikipedia: Vektorfont https://de.wikipedia.org/wiki/Vektorfont.* (Besucht am 11. 11. 2019).

[44] *Wikipedia: Vektorisierung.* URL: https://de.wikipedia.org/wiki/Vektorisierung_(Grafik) (besucht am 11. 11. 2019).

[45] *Wikipedia: Bildsynthese https://de.wikipedia.org/wiki/Bildsynthese.* (Besucht am 11. 11. 2019).

[46] *Wikipedia: Skalierung(Computergrafik).* URL: https://de.wikipedia.org/wiki/Skalierung_(Computergrafik) (besucht am 11. 11. 2019).

[47] *Wikipedia: Responsive Webdesign.* URL: https://de.wikipedia.org/wiki/Responsive_Webdesign (besucht am 11. 11. 2019).

[48] *Youtube-Video zur komplexen Fourieranalyse.* URL: https://www.youtube.com/watch?v=r6sGWTCMz2k (besucht am 09. 11. 2019).

[49] *Einführung in die Begriffe der Gaußschen Zahlenebene.* URL: https://www.mathe-online.at/materialien/Andreas.Pester/files/ComNum/inhalte/zahlenebene.html (besucht am 11. 11. 2019).

[50] *Wikipedia: Poi.* URL: https://de.wikipedia.org/wiki/Poi (besucht am 11. 11. 2019).

[51] *Artikel zu Poi-Terminologie.* URL: https://www.homeofpoi.com/en/lessons/teach/POI/Poi-terminology/Poi-terminology (besucht am 09. 11. 2019).

[52] Alexander Bobenko. *Differentialgeometrie von Kurven und Flächen.* URL: http://page.math.tu-berlin.de/~bobenko/Lehre/Skripte/KuF.pdf (besucht am 09. 11. 2019).

[53] *Definiton Physics Engine.* URL: https://gamedev.stackexchange.com/questions/129686/what-exactly-is-a-physics-engine (besucht am 08. 11. 2019).