Informatik
Hauptcampus

H O C H
S C H U L E
T R I E R

Fortschrittliche Textursynthese mithilfe von Image-Quilting-Techniken

Modul: Tools- und Pluginprogrammierung

Betreuer: Prof. Dr. Christof Rezk-Salama, Dipl.-Inf.

Bearbeiter: Mohammed Salih Mezraoui

31.08.2024

Abstrakt

Das Ziel dieses Projekts ist die Entwicklung einer Methode zur Erzeugung großer, visuell kohärenter Texturen durch nahtlose Integration kleinerer Texturfelder, die aus einem Eingabebild entnommen werden – ein Verfahren, das als Image Quilting bekannt ist. Diese Technik stellt sicher, dass die resultierenden Texturen nicht nur visuell ansprechend sind, sondern auch frei von auffälligen Nähten oder wiederholenden Mustern. Durch den Einsatz fortschrittlicher Algorithmen wie dem Minimum Error Boundary Cut und der Mean Squared Error Blockauswahl wird die Ausrichtung der Texturfelder optimiert, um das natürliche Erscheinungsbild der Textur zu bewahren. Das Projekt untersucht auch die Flexibilität des Quiltverfahrens, indem Variationen in Blockgröße, Überlappung und Toleranzgrenzen ermöglicht werden, um eine vielfältige Palette von Texturen zu erzeugen. Diese Methode hat potenzielle Anwendungen in Bereichen wie Computergrafik, Game Design und digitaler Kunst, in denen hochwertige Texturen unerlässlich sind.

Inhaltsverzeichnis

Kapitel 1

Grundlagen und Methoden

Dieses Kapitel führt in die grundlegenden Konzepte und Techniken der Textursynthese mittels Image Quilting ein. Die Entscheidung, sich mit diesem Thema zu befassen, basiert auf der zunehmenden Relevanz nahtloser Texturen in der Computergrafik, insbesondere in der Spieleentwicklung und im digitalen Design. Darüber hinaus wird die Bedeutung der zugrunde liegenden Algorithmen für die Texturverarbeitung sowohl aus theoretischer als auch aus praktischer Sicht beleuchtet.

1.1 Gründe für die Themenwahl

Die Entscheidung für das Thema ” Image-Quilting” wurde durch mehrere Schlüssel-faktoren beeinflusst. Die Textursynthese spielt eine entscheidende Rolle in der Computergrafik, insbesondere in Bereichen wie Spieleentwicklung, digitalem Design und visuellen Effekten, wo die Erstellung nahtloser und optisch ansprechender Texturen für realistische Umgebungen unerlässlich ist.

Darüber hinaus bietet das Thema eine tiefgehende Auseinandersetzung mit den Algorithmen hinter der Texturverarbeitung. Image Quilting, wie von Efros und Freeman vorgeschlagen, stellt eine effektive Methode dar, um aus kleinen Mustern große, kohärente Texturen zu erzeugen. Diese Technik ist sowohl theoretisch interessant als auch praktisch relevant [1].

Schließlich ermöglicht das Projekt praktische Erfahrungen in der Implementierung komplexer Algorithmen und der Bewertung ihrer Leistung in realen Anwendungen. Dies stellt nicht

nur eine intellektuelle Herausforderung dar, sondern fördert auch die Entwicklung wertvoller technischer Fähigkeiten, die in der Technologiebranche sehr gefragt sind.

1.2 Image Quilting: Eine Technik zur exemplar-basierten Textursynthese

Image Quilting ist eine Technik zur Exemplar-basierten Textursynthese, die neue Texturen aus einem Eingabemuster generiert. Dabei werden Patches aus der Eingabetextur zusammengenäht, um eine größere Ausgabetextur zu erstellen. Der Prozess umfasst die Auswahl kompatibler Patches basierend auf Überlappungsbereichen und die Berechnung eines optimalen Randzuschnitts, um sichtbare Nähte zu minimieren. Diese Methode reproduziert effektiv Makro-Texturen und erhält dabei die visuelle Kohärenz sowie die statistischen Eigenschaften der ursprünglichen Textur [1].

1.3 Schritte im Image-Quilting-Prozess

1. **Abtastung**:
   Der Image-Quilting-Prozess beginnt mit der Auswahl eines Patches aus der Eingabetextur. Dieser Patch wird so gewählt, dass er zum Überlappungsbereich der teilweise definierten Textur passt, die als alter Patch bezeichnet wird. Der Algorithmus sucht nach einem neuen Patch, der den Pixelwerten in diesem Überlappungsbereich möglichst ähnlich ist, um visuelle Kohärenz zu gewährleisten. Obwohl die Auswahl zufällig erfolgt, wird sie durch die Notwendigkeit der Ähnlichkeit im Überlappungsbereich geleitet, um sicherzustellen, dass der neue Patch nahtlos in die vorhandene Textur integriert wird [1].

2. **Überlappung**:
   Nachdem der neue Patch ausgewählt wurde, definiert der Algorithmus einen Überlappungsbereich, in dem der alte und der neue Patch verbunden werden. Diese Überlappung ist entscheidend für eine reibungslose Verbindung der Patches. Der Überlappungsbereich wird sorgfältig dimensioniert, um einen nahtlosen Übergang zwischen den Patches zu ermöglichen und gleichzeitig das Risiko sichtbarer Nähte zu minimieren. Das Ziel des Algorithmus ist es, sicherzustellen, dass dieser Bereich den neuen Patch effektiv in die bestehende Textur integriert und dabei ein kontinuierliches und natürliches Erscheinungsbild bewahrt [1].

3. **Berechnung des minimalen Schnitts**:

   Der letzte Schritt besteht darin, die optimale Schnittgrenze zwischen den Überlappungsbereichen des alten und neuen Patches zu berechnen. Der Algorithmus bestimmt diese Grenze, indem er einen Pfad findet, der visuelle Diskontinuitäten bei der Kombination der Patches minimiert. Oft wird ein linearer Programmierungsansatz verwendet, um diesen optimalen Schnitt zu berechnen und den Übergang so glatt wie möglich zu gestalten. Das Ergebnis ist ein neuer Patch, der sich harmonisch mit dem alten Patch entlang der berechneten Schnittgrenze verbindet und eine visuell kohärente und ästhetisch ansprechende Textur im Ausgabe-Bild erzeugt [1].

Kapitel 2

Projektstruktur

Dieses Kapitel bietet einen Überblick über die Struktur des Projekts und die verwendeten Technologien, insbesondere die Vorteile von C++ für die Implementierung der Image-Quilting-Technik. Es werden die Hauptmerkmale und die Verzeichnisstruktur des Projekts erläutert, um ein besseres Verständnis für die Organisation und die Kernkomponenten des Codes zu vermitteln.

2.1 Hauptmerkmale und Vorteile der Nutzung von C++ für Image Quilting

C++ ist eine ideale Wahl für dieses Projekt aufgrund seiner Leistung und Flexibilität. Es bewältigt rechenintensive Aufgaben wie Sampling, Überlappung und die Berechnung des minimalen Schnitts effizient. Die fortschrittliche Speicherverwaltung von C++ sorgt für eine reibungslose Verarbeitung großer Texturdaten, während die objektorientierten Funktionen eine modulare und wartbare Code-Struktur ermöglichen. Die umfangreiche Standardbibliothek der Sprache, zusammen mit Drittanbieter-Bibliotheken, unterstützt die Bildverarbeitung und mathematische Berechnungen. Zudem gewährleistet die plattformübergreifende Fähigkeit von C++, dass der Code nahtlos auf verschiedenen Systemen läuft. Die umfangreiche Community und die zahlreichen Ressourcen von C++ tragen zusätzlich zur Optimierung und Problemlösung bei, was C++ zu einem leistungsstarken Werkzeug für effizientes und skalierbares Image Quilting macht [2].

2.2 Struktur der Projektdatei und Erläuterung des Codes

Die folgenden Unterabschnitte geben einen Überblick über die Hauptkomponenten des Projekts, einschließlich der Verzeichnisstruktur, der Header-Dateien, die wichtige Schnittstellen definieren, und der Quellcodedateien, die die Kernfunktionalitäten implementieren.

2.2.1 Struktur der Projektdatei

Dieses Projekt ist in mehrere wichtige Verzeichnisse unterteilt:

- **include/**: Enthält Header-Dateien (`ErrorMetrics.h`, `SeamCarving.h`, `TextureQuilter.h`), die die zentralen Schnittstellen definieren.
- **src/**: Enthält die Quellcode-Dateien (`ErrorMetrics.cpp`, `SeamCarving.cpp`, `TextureQuilter.cpp`, `main.cpp`), die die Hauptalgorithmen und die Programmlogik implementieren.
- **gallery/**: Enthält Eingabe- und Ausgabebilder zur Demonstration des Image-Quilting-Prozesses.
- **CMakeLists.txt**: Verwaltet den Build-Prozess über CMake.

2.2.2 Header-Dateien

1. **`TextureQuilter.h`**: Diese Datei definiert die Klasse `TextureQuilter`, die den Image-Quilting-Algorithmus implementiert. Sie ermöglicht das Setzen von Kachel- und Nahtparametern und bietet Optionen für MSE-Auswahl und minimale Schnittnähte. Die Hauptfunktion, `quiltTexture`, erzeugt eine neue Textur, indem sie Patches aus einer Quelltextur zusammensetzt.
2. **`SeamCariving.h`**: Diese Datei enthält Funktionalitäten zur Berechnung von minimalen Kosten-Nähten, um die beste Naht für das Quilten von Texturen zu finden. Sie definiert die Struktur `MinCostCutData` und zwei Funktionen, `findVerticalMinCostSeam` und `findHorizontalMinCostSeam`, die vertikale und horizontale Nähte mit minimalen Kosten ermitteln.
3. **`ErrorMetrics.h`**: Diese Datei definiert Funktionen zur Berechnung von Fehlermaßen zwischen Texturen. Sie enthält die Funktionen `calculateErrorSum` zur Berechnung der

Fehler-Summe und `calculateMSE` zur Berechnung des mittleren quadratischen Fehlers (MSE), die beide zur Bewertung der Übereinstimmung von Textur-Patches dienen.

2.2.3 Quelldateien

- **`TextureQuilter.cpp`**: Diese Quellcodedatei implementiert den Image-Quilting-Algorithmus, der in `TextureQuilter.h` deklariert ist. Sie enthält Funktionen zur Auswahl und Platzierung von Texturpflastern sowie zur Berechnung von Nähten mit minimalen Fehlern, um nahtlose Texturen zu erzeugen.
- **`SeamCarving.cpp`**: Diese Quellcodedatei implementiert Algorithmen zur Berechnung von vertikalen und horizontalen Nähten mit minimalen Kosten, wie in `SeamCarving.h` deklariert. Die Hauptfunktionen `findVerticalMinCostSeam` und `findHorizontalMinCostSeam` bestimmen die günstigsten Nähte für das nahtlose Verbinden von Texturen.
- **`ErrorMetrics.cpp`**: Diese Quellcodedatei implementiert Funktionen zur Berechnung von Fehlermaßen zwischen Texturen, wie sie in `ErrorMetrics.h` deklariert sind. Die Funktionen `calculateErrorSum` und `calculateMSE` bewerten die Übereinstimmung und den Fehler zwischen Texturpflastern, um die Qualität des Quilting-Ergebnisses zu bestimmen.
- **`main.cpp`**: Diese Quellcodedatei enthält den Haupteinstiegspunkt des Programms. Sie implementiert das Parsing der Kommandozeilenargumente, die Initialisierung des Image-Quilting-Prozesses und das Speichern des generierten Texturergebnisses in verschiedenen Bildformaten.

Kapitel 3

Ergebnisse und Analyse

In diesem Kapitel wird beschrieben, wie das Projekt ausgeführt wird, einschließlich der notwendigen Schritte zur Kompilierung und Ausführung des Programms. Darüber hinaus werden verschiedene Ergebnisse vorgestellt, wobei sowohl Eingabebilder als auch die daraus generierten Ausgabe-Texturen gezeigt werden, um die Funktionsweise und Effektivität des Image-Quilting-Algorithmus zu demonstrieren.

3.1 Projekt ausführen und Parameterbeschreibung

Um das Projekt auszuführen, ist es notwendig, sich im Verzeichnis `cmake-build-debug` zu befinden und dann den folgenden Befehl auszuführen:

```
./ImageQuilting_main --input ../gallery/input6.png \
                     --output ../gallery/output7.png \
                     --width 500 --height 500 \
                     --tileW 32 --tileH 32 \
                     --seamW 4 --seamH 4 \
                     --mseSelect true --minCut true \
                     --tolerance 0.1
```

Dieser Befehl führt das Programm `ImageQuilting_main` aus und erzeugt eine Ausgabe-Textur basierend auf den angegebenen Parametern. Die wichtigsten Parameter sind:

- `--input`: Gibt das Eingabebild an, das gequiltet werden soll, in diesem Fall `input6.png`

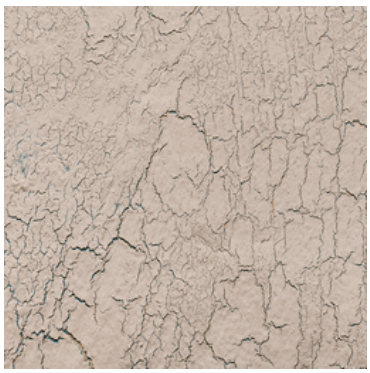
im `gallery`-Verzeichnis.

- `--output`: Legt den Namen und Pfad für das generierte Ausgabebild fest, hier `output7.png` im `gallery`-Verzeichnis.
- `--width` und `--height`: Bestimmen die Abmessungen der erzeugten Ausgabe-Textur in Pixeln (500x500).
- `--tileW` und `--tileH`: Bestimmen die Breite und Höhe der Texturkacheln (jeweils 32 Pixel).
- `--seamW` und `--seamH`: Legen die Breite und Höhe der Nähte fest, die beim Quilten minimiert werden (jeweils 4 Pixel).
- `--mseSelect`: Aktiviert die Auswahl von Kacheln basierend auf dem mittleren quadratischen Fehler (MSE) (`true` bedeutet aktiviert).
- `--minCut`: Aktiviert den minimalen Schnitt für die Nähte (`true` bedeutet aktiviert).
- `--tolerance`: Bestimmt die Toleranz für die Auswahl der Kacheln basierend auf dem MSE (hier 0.1).

Dieser Befehl ermöglicht die Erstellung einer nahtlosen, gequilteten Textur unter Verwendung der spezifizierten Eingabedaten und Parameter.

3.2 Beispiele

In diesem Unterabschnitt werden die Eingabebilder und die daraus generierten Ausgabe-Texturen dargestellt. Jedes Eingabebild wird auf der linken Seite gezeigt, während die entsprechende Ausgabe-Textur auf der rechten Seite angezeigt wird.



Kapitel 4

Zusammenfassung

Dieses Projekt befasst sich mit der Entwicklung einer Methode zur Erstellung großer, visuell kohärenter Texturen mithilfe der Technik des Image Quilting. Die Methode ermöglicht die nahtlose Integration kleinerer Texturblöcke aus einem Eingabebild, sodass die resultierenden Texturen frei von sichtbaren Nähten oder sich wiederholenden Mustern sind. Der Prozess beginnt mit der Auswahl eines neuen Patches, der optisch zu einem bestehenden Patch passt, gefolgt von der Definition eines Überlappungsbereichs, um eine nahtlose Übergangsstelle zu gewährleisten. Schließlich wird die optimale Schnittgrenze zwischen den Patches berechnet. Das Projekt untersucht auch die Flexibilität des Verfahrens, indem Variationen in Blockgröße, Überlappung und Toleranzgrenzen ermöglicht werden, um eine Vielzahl von Texturen zu erstellen. Die potenziellen Anwendungen dieser Technik sind vielfältig und reichen von Computergrafik über Game Design bis hin zur digitalen Kunst, wo hochwertige Texturen von entscheidender Bedeutung sind.

Literaturverzeichnis

[1] Lara Raad Cisa and Bruno Galerne. “Efros and Freeman Image Quilting Algorithm for Texture Synthesis”. In: *Image Processing On Line* 7 (Jan. 2017), pp. 1–22. DOI: 10.5201/ipol.2017.171.

[2] cppreference contributors. *C++ Reference*. Accessed: 2024-07-12. 2024. URL: https://en.cppreference.com/w/.