

**Felipe Vieira Cortes**

**An alternative formalization of reference typing**

**Tese de Doutorado**

Thesis presented to the Programa de Pós-graduação em Informática, do Departamento de Informática da PUC-Rio in partial fulfillment of the requirements for the degree of Doutor em Informática.

Advisor: Prof. Roberto Ierusalimsky

Rio de Janeiro  
March 2023



**Felipe Vieira Cortes**

## **An alternative formalization of reference typing**

Thesis presented to the Programa de Pós-graduação em Informática da PUC-Rio in partial fulfillment of the requirements for the degree of Doutor em Informática. Approved by the Examination Committee:

**Prof. Roberto Ierusalimsky**

Advisor

Departamento de Informática – PUC-Rio

**Prof. Roberto Ierusalimsky**

Departamento de Informática – PUC-Rio

Rio de Janeiro, March 11th, 2023

All rights reserved.

**Felipe Vieira Cortes**

Graduated in computer science by the Pontifícia Universidade Católica do Rio de Janeiro.

Bibliographic data

Cortes, Felipe V.

An alternative formalization of reference typing / Felipe Vieira Cortes; advisor: Roberto Ierusalimsky. – 2023.

22 f: il. color. ; 30 cm

Tese (doutorado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática, 2023.

Inclui bibliografia

1. Informática – Teses. 2. Type References. 3. Programming Languages. 4. Formalization. I. Ierusalimsky, Roberto. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. III. Título.

CDD: 004

## Acknowledgments

## Abstract

Cortes,Felipe V.; Ierusalimschy, Roberto (Advisor). **An alternative formalization of reference typing**. Rio de Janeiro, 2023. 22p. Tese de Doutorado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Reference Typing is a ...

## Keywords

Type References; Programming Languages; Formalization.

## Resumo

Cortes, Felipe V.; Ierusalimschy, Roberto. **An alternative formalization of reference typing**. Rio de Janeiro, 2023. 22p. Tese de Doutorado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

A tipagem de referencia eh ...

## Palavras-chave

Type References; Programming Languages; Formalization.

## **Table of contents**

<b>1</b>	<b>Introduction</b>	<b>14</b>
<b>2</b>	<b>Previous Work</b>	<b>15</b>
<b>3</b>	<b>Proposal</b>	<b>18</b>
<b>4</b>	<b>Results</b>	<b>20</b>
4.1	Comparison	20
<b>5</b>	<b>Conclusion and future work</b>	<b>21</b>
<b>6</b>	<b>Bibliography</b>	<b>22</b>

## List of figures

Figure 2.1	A set of three subfigures: (a) describes the first subfigure; (b) describes the second subfigure; (c) describes the third subfigure.	15
(a)	Bamboo-pile Vertically Inserted Position	15
(b)	Bamboo-pile Normal Inserted Position	15
(c)	bamboo-pile Inserted 45° angle	15
Figure 2.2	A set of six subfigures in two pages.	16
(a)	Bamboo-pile Vertically Inserted Position	16
(b)	Bamboo-pile Normal Inserted Position	16
(c)	bamboo-pile Inserted 45° angle	16
(d)	Bamboo-pile Vertically Inserted Position	17
(e)	Bamboo-pile Normal Inserted Position	17
(f)	bamboo-pile Inserted 45° angle	17



## List of tables

Table 4.1	Results for devil mesh	20
-----------	------------------------	----

## List of algorithms

Algorithm 1	Escolha das amostras iniciais	19
-------------	-------------------------------	----

**List of codes**

Code 1	Mean Filter	18
Code 2	Mean Filter	21

## List of Abbreviations

ADI – Análise Digital de Imagens

BIF – *Banded Iron Formation*

*My beautifull epigraph*

**Wassily Kandinsky**, *Regards sur le passé.*

# 1

## Introduction

- What is Reference typing?
- What is the main representation (from Pierce)?
- What we want to propose studying?
- What is our alternative representation?
- What are the differences
- Why should we formalize an idea?
- Which Theorems we want to prove?
- Which usabilites can we identify by this formalization?
- Is this representation simpler or more complicated to prove?
- Is this dissertation meant to be didactic?

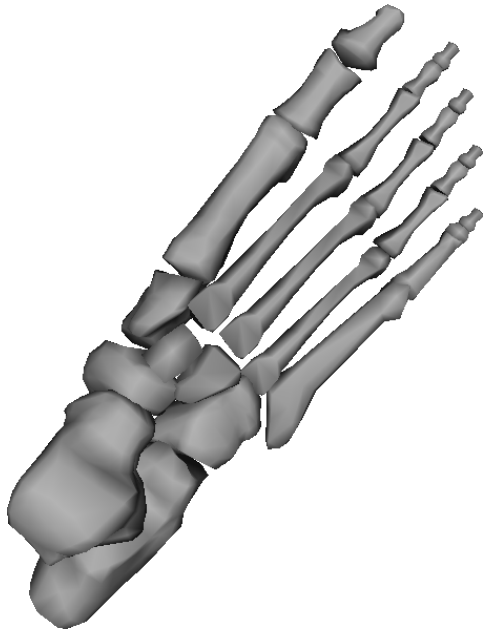
Type Reference is a technique of representing the types of a program in a store. Some properties of this store can be defined to verify the correctness of a given representation. Formalizing these concepts makes it more understandable, in a way which computer scientists can communicate their ideas through a standardized set of rules. Why coq? Coq is a expressive functional programming language used for stating and proving logical assertions and a standard tool for researchers to reason about complex language definitions (PIERCE et al., 2022)

This document is structured as follows. In Chapter 2 we present some previous work relevant to our problem. In Chapter 3 we explain our proposal. In Chapter 4 we show our results. Finally, in Chapter 5 we present our conclusion and future work.

## 2

### Previous Work

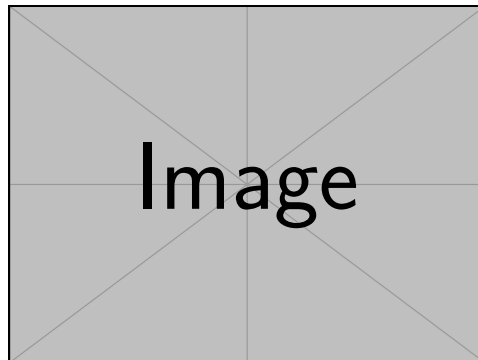
Early smoothing methods tried to minimize... In the figure 2.2d we see...



(a) Bamboo-pile Vertically Inserted Position

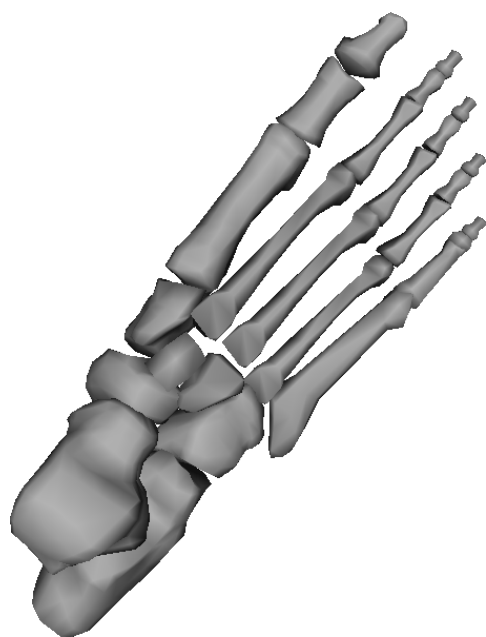


(b) Bamboo-pile Normal Inserted Position

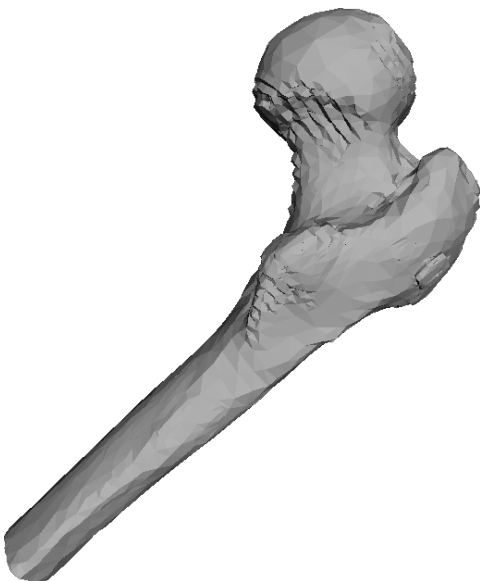


(c) bamboo-pile Inserted 45° angle

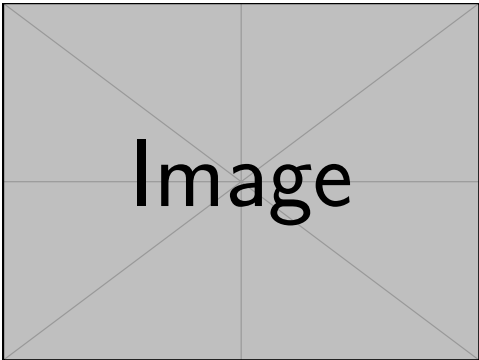
Figure 2.1: A set of three subfigures: (a) describes the first subfigure; (b) describes the second subfigure; (c) describes the third subfigure.



(a) Bamboo-pile Vertically Inserted Position



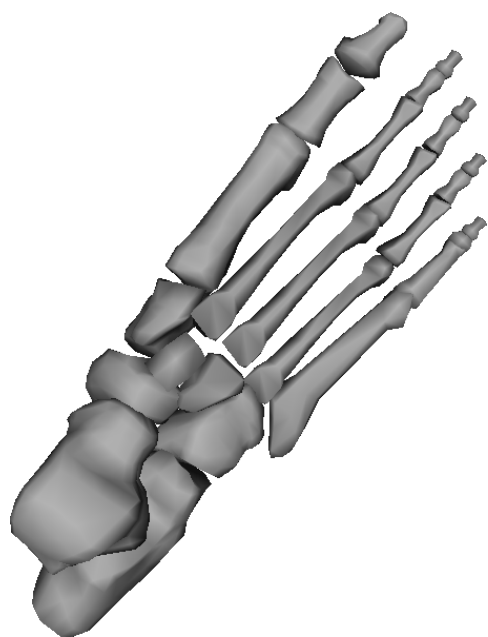
(b) Bamboo-pile Normal Inserted Position



(c) bamboo-pile Inserted 45° angle

Figure 2.2: A set of six subfigures in two pages.

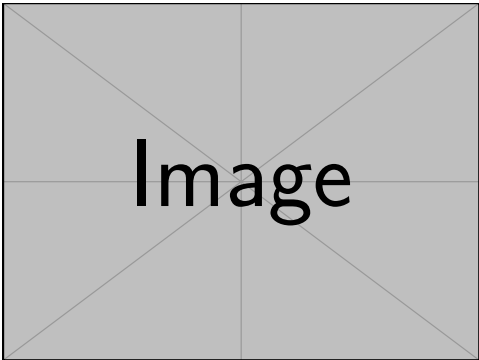




(d) Bamboo-pile Vertically Inserted Position



(e) Bamboo-pile Normal Inserted Position



(f) bamboo-pile Inserted 45° angle

Figure 2.2: A set of six subfigures in two pages.(Continuation)

### 3 Proposal

Equation example 1:

$$\begin{aligned} \min_u \int_{x_i \in X} \int_{x_j \in X} q_{ij} u_i u_j da + \int_{x_i \in X} \|x' - x_i\| u_i da \\ s.t. \quad u \in [0, 1] \quad \wedge \quad \int_{x_i \in X} u da = a_0, \end{aligned} \quad (3-1)$$

Equation exmaple 2:

$$\begin{aligned} \min_{\mathbf{u}} \alpha \mathbf{u}^T \mathbf{A}^T \mathbf{Q} \mathbf{A} \mathbf{u} + \beta \mathbf{d}^T \mathbf{a}' \mathbf{A} \mathbf{u} + \gamma \mathbf{u}^T \mathbf{G}^T \mathbf{G} \mathbf{u} + \delta \mathbf{f}^T \mathbf{a}' \mathbf{A} \mathbf{u} \\ s.t. \quad \mathbf{0} \leq \mathbf{u} \leq \mathbf{1} \wedge \mathbf{a}^T \mathbf{u} = a_0. \end{aligned} \quad (3-2)$$

Equation example 3:

$$\mathbf{G} = (g_{ij}) = \begin{cases} \sum_{f_k \in N_f(f_i)} l_{ik} & i = j \\ -l_{ij} & e_{ij} \in E \\ 0 & \text{otherwise} \end{cases} \quad (3-3)$$

---

#### Code 1: Mean Filter

---

```

1 #
   -----#

2 # Create filter function
3 # l is the width of window
4 #
   -----#

5 meanfilter <- function( l, imagem ) {
6   if( l%%2 == 0 )
7     print("Please, type an odd number!")
8   imagem.result <- imagem
9   lp1d2 <- (l-1)/2
10  L <- dim(imagem)[1]
11  C <- dim(imagem)[2]
12  for( j in as.integer(lp1d2+1) : as.integer(C-lp1d2)) {
13    for( i in as.integer(lp1d2+1) : as.integer(L-lp1d2)) {
14      imagem.result[i,j] <- mean(imagem[as.integer(i-lp1d2):as.
        integer(i+lp1d2), as.integer(j-lp1d2):as.integer(j+lp1d2)
        ])
15    }
16  }

```

```

17  print("Image filtered with success!")
18  return(imagem.result)
19 }
20 #
    -----#
21 # End of Script.
22 #
    -----#

```

---

**Algorithm 1:** Escolha das amostras iniciais

---

**Input:** Malha e quantidade de pontos a ser amostrado

**Output:** Pontos amostrados na malha

- 1 *Crie um vetor de números randômicos entre  $[0, 1]$  com a quantidade de pontos a ser amostrada e ordene-o*
  - 2 *Calcule a área total dos triângulos da malha*
  - 3 **for**  $i = 0$  **to**  $\text{numeroDePontos}$  **do**
  - 4     *Navegue entre as faces acumulando a sua  $\frac{\text{area}}{\text{areaTotal}}$  até achar a face com valor acumulado  $\geq \text{numerosRandomicos}[i]$*
  - 5     *Pegue um ponto randômico dentro da face utilizando o método de Turk e adicione no vetor do resultado*
-

# 4

## Results

Table example. Table 4.1 shows results.

Table 4.1: Results for devil mesh

	Mean Vertex Dis- tance	L2 Vertex Based	Mean Quadric	MSAE	L2 Nor- mal Based	Tangential	Mean Discrete Curva- ture	Area Error	Volume Error
(FLEISHMAN; DRORI; COHEN- OR, 2003) (JONES; DU- RAND; DES- BRUN, 2003) (SUN et al., 2007) (ZHENG et al., 2011) (HE; SCHAE- FER, 2013) (ZHANG et al., 2015) (YADAV; REITE- BUCH; POLTH- IER, 2016) Ours	0.061277	0.110973	0.236219	19.697900	0.055170	0.047678	0.090284	0.051443	0.045645
	0.001293	0.002800	0.002289	21.237300	0.021589	0.013026	0.087991	0.000364	0.002621
	0.001439	0.002880	0.003540	14.043200	0.012654	0.008911	0.055849	0.007806	0.000582
	0.000713	0.001537	0.001824	12.171400	0.009654	0.005781	0.054567	0.005617	0.000425
	0.002531	0.004560	0.007108	13.830100	0.017459	0.010314	0.114528	0.001686	0.001786
	0.001623	0.003079	0.005048	10.454200	0.015233	0.008054	0.094668	0.002629	0.001326
	0.000737	0.001548	0.001493	16.880800	0.014129	0.006974	0.079952	0.000209	0.002375
	0.000987	0.001902	0.002686	11.574200	0.010632	0.006796	0.075106	0.003970	0.000722

## 4.1

### Comparison

## 5

### Conclusion and future work

We proposed an algorithm for triangular mesh denoising with detail preservation...

---

#### Code 2: Mean Filter

---

```
1 #
   -----#

2 # Create filter function
3 # l is the width of window
4 #
   -----#

5 meanfilter <- function( l, imagem ) {
6   if( l%%2 == 0 )
7     print("Please, type an odd number!")
8   imagem.result <- imagem
9   lp1d2 <- (l-1)/2
10  L <- dim(imagem)[1]
11  C <- dim(imagem)[2]
12  for( j in as.integer(lp1d2+1) : as.integer(C-lp1d2)) {
13    for( i in as.integer(lp1d2+1) : as.integer(L-lp1d2)) {
14      imagem.result[i,j] <- mean(imagem[as.integer(i-lp1d2):as.
15                                integer(i+lp1d2), as.integer(j-lp1d2):as.integer(j+lp1d2)
16                                ])
17    }
18  }
19  print("Image filtered with success!")
20  return(imagem.result)
21 #
   -----#

21 # End of Script.
22 #
   -----#
```

---

## 6

### Bibliography

FLEISHMAN, S.; DRORI, I.; COHEN-OR, D. Bilateral mesh denoising. In: ACM. **ACM transactions on graphics (TOG)**. [S.l.], 2003. v. 22, n. 3, p. 950–953. Cited in page 20.

HE, L.; SCHAEFER, S. Mesh denoising via  $l_0$  minimization. **ACM Transactions on Graphics (TOG)**, ACM, v. 32, n. 4, p. 64, 2013. Cited in page 20.

JONES, T. R.; DURAND, F.; DESBRUN, M. Non-iterative, feature-preserving mesh smoothing. In: ACM. **ACM Transactions on Graphics (TOG)**. [S.l.], 2003. v. 22, n. 3, p. 943–949. Cited in page 20.

PIERCE, B. C. et al. **Logical Foundations**. [S.l.]: Electronic textbook, 2022. v. 1. (Software Foundations, v. 1). Version 6.2, URL-<http://softwarefoundations.cis.upenn.edu>. Cited in page 14.

SUN, X. et al. Fast and effective feature-preserving mesh denoising. **IEEE transactions on visualization and computer graphics**, IEEE, v. 13, n. 5, p. 925–938, 2007. Cited in page 20.

YADAV, S.; REITEBUCH, U.; POLTHIER, K. Mesh denoising based on normal voting tensor and binary optimization. **arXiv preprint arXiv:1607.07427**, 2016. Cited in page 20.

ZHANG, W. et al. Guided mesh normal filtering. In: WILEY ONLINE LIBRARY. **Computer Graphics Forum**. [S.l.], 2015. v. 34, n. 7, p. 23–34. Cited in page 20.

ZHENG, Y. et al. Bilateral normal filtering for mesh denoising. **IEEE Transactions on Visualization and Computer Graphics**, IEEE, v. 17, n. 10, p. 1521–1530, 2011. Cited in page 20.