UNIVERSITAT JAUME I

Grado en Matemática Computacional

Estancia en Prácticas y Proyecto Final de Grado

Estimación de curvatura en curvas discretas

Autor:
Alexis Turch González

Supervisor:
Rafael Forcada Martínez
Tutor académico:
Vicente Cervera Mateu

Fecha de lectura: 09 de julio de 2018
Curso académico 2017/2018

Resumen

El presente Trabajo Final de Grado abarca los aspectos más importantes de mi estancia en prácticas en Actualmed así como un análisis sobre las distintas posibles curvaturas en el ámbito de la Geometría Diferencial Discreta.

En la empresa tuve como objetivos el lograr elaborar una herramienta de análisis en base a las imágenes médicas con las que se trataba allí. De esta forma se irá desarrollando mi estancia en prácticas, detallando los objetivos del proyecto así como la metodología utilizada para lograr dichas metas.

A continuación, en cuanto al tema para desarrollar en la memoria se ha escogido la Geometría Diferencial Discreta. Para ello se irán definiendo términos básicos de la geometría diferencial para posteriormente extrapolarlo al ámbito discreto.

De esta forma se verá como cuando se pasa al entorno discreto, la materia se torna más compleja de entender y aplicar. Observaremos como la definición de curvatura en el entorno continuo es algo fácil e intuitivo de entender y formular mientras que en el discreto habrá que desarrollarlo con más cuidado.

Palabras clave

Actualmed, DICOM, Geometría Diferencial Discreta, DDG, curvatura.

Keywords

Actualmed, DICOM, Discrete Differential Geometry, DDG, curvature.

Índice general

Capítulo 1

Introducción

1.1. Contexto y motivación del proyecto

El presente *Trabajo Final de Grado* (TFG) se centra en el desarrollo de mi estancia en prácticas en la empresa Actualmed, localizada en el edificio de Espaitec de la UJI. Decidí escoger un proyecto de trabajo relacionado con el tratamiento de imágenes pues es un ámbito meramente geométrico, una de las ramas matemáticas más interesantes.

Los objetivos principales a alcanzar en Actualmed era conseguir realizar un tratamiento específico con las imágenes con las que se trabajaba allí desde un punto de vista más informático. Concretando, mis acciones se centraban en conocer como se manejan las imágenes médicas y hacer un posterior tratado de los datos que contenían para poder mostrar los más relevantes mediante una herramienta de análisis.

Sirviéndome de las tan elaboradas imágenes médicas, determino centrarme en el tratamiento de imágenes desde un punto de vista más matemático. En el contexto docente se centran en desarrollar la teoría a partir de un entorno continuo, en el cual todo lo que ocurre suele ser más intuitivo y fácil de entender y explicar. En cambio, vamos a aplicar criterios de dicha materia ya conocidas para aplicarlo al caso discreto.

Para ello, lo que haremos será dar una serie de definiciones básicas en la geometría diferenciable y a partir de ahí deducir los valores de dichos conceptos. En concreto, nos centraremos en la correcta explicación de la curvatura, puesto que en el ámbito plano todo se reduce a una definición e interpretación de pocas líneas, mientras que en el ámbito continuo se complica. Si extrapoláramos directamente la definición de curvatura continuo a curvatura discreta, los conceptos empiezan a fallar y perderse.

Llegados a este punto, ahondaremos en una nueva definición que no solo conservará y aproximará las nociones establecidas en los entornos diferenciables sino que además se obtendrán nuevas y más amplias propiedades e interpretaciones. Es de esta forma como la curvatura en el ámbito discreto forma una entidad coherente por sí misma, es decir, que no necesita de términos continuos para definirse y completarse.

En síntesis, analizaremos e investigaremos el tratamiento de imágenes desde el punto de vista informático y nos centraremos en las nociones más básicas sobre representación de imágenes desde el punto de vista matemático.

Capítulo 2

Estancia en prácticas

En esta sección se explica el contexto del proyecto, comenzando por describir la empresa en la que se lleva a cabo al igual que también se explica el ámbito en el que se desarrolla. Además, se establece el alcance del proyecto, las funciones que se realizan y los beneficios que aportan a la empresa.

Antes de comenzar, me gustaría añadir una serie de definiciones que ayuden a la comprensión del texto:

- **MySQL** es un sistema de gestión de bases de datos relacional considerada como la base datos de código abierto más popular del mundo para entornos de desarrollo web.
- **phpMyAdmin** es una herramienta escrita en PHP con la intención de manejar la administración de MySQL a través de páginas web utilizando Internet.
- **MongoDB** es una base de datos NoSQL orientada a documentos, es decir, en lugar de guardar los datos en registros, guarda los datos en documentos.
- **DICOM** (Digital Imaging and Communications in Medicine) es un protocolo estándar de comunicación entre sistemas de información y a la vez un formato de almacenamiento de imágenes médicas que aparece como solución a los problemas de interoperabilidad entre tipos de dispositivos.
- **Diccionario Python**: estructuras de datos que permite guardar un conjunto no ordenado de pares clave-valor, siendo las claves únicas dentro de un mismo diccionario (es decir que no pueden existir dos elementos con una misma clave). Por ejemplo: diccionario = {clave1: valor1, clave2: valor2}

- **Listas Python**: estructura de datos que se representa como una secuencia de valores encerrados entre corchetes y separados por comas. Por ejemplo: lista = [“dato1”,“dato2”,“dato3”].
- **Lista de diccionarios Python**: Por ejemplo: listaDiccionarios = [{clave1: valor1, clave2: valor2}, {clave1: valor1, clave2: valor2}]
- **Flask** es un framework escrito en Python que permite crear aplicaciones web.

2.1. Objetivos del proyecto

Actualmed es una empresa especializada en teleradiología que maneja *Actualpacs*, una plataforma Secure Cloud de archivado y gestión de imagen médica que permite acceder a los estudios a la máxima velocidad, además de informar desde cualquier lugar y dispositivo, permitiendo así una comunicación más rápida con los radiólogos para la transmisión de estudios e informes. Dicha plataforma dispone de una base de datos de más de 250.000 artículos científicos. [1]

A lo largo de mi estancia en dicha empresa, tengo como meta el conseguir crear una **herramienta de análisis** en base a la información extraída de estudios e informes médicos radiológicos. El motivo principal de la empresa por querer tener esta herramienta accesible en cualquier momento es debido al hecho de que puede ayudar tanto a tomar decisiones empresariales como a la posibilidad de extraer modelos de predicción a Actualmed.

A grandes rasgos, mis verdaderos objetivos son el afrontar problemas de la vida cotidiana de tal forma que sea capaz de resolverlos con las competencias aprendidas durante mi enseñanza junto con mi propia capacidad de autoaprendizaje y confrontamiento de contrariedades. Una persona no nace aprendida al igual que un universitario a las puertas de acabar un grado no se saca el título sabiéndolo todo. Es a la hora de la verdad, es decir, en el momento de incertidumbre en el que te encuentras al entrar en el mundo empresarial donde verdaderamente se aprende. Siempre y cuando cuentes con la motivación exigida en esos momentos de perdición a la vez de tener la posibilidad de contar con compañeros que tengan la predisposición de ayudarte en todo lo que te sea necesario.

2.2. Metodología y definición de tareas

Mi tarea a realizar será la de extraer la información más destacable de la base de datos relacional de la que disponen en *MySQL* (llamada **pacsdb**) e importarla a una base de datos no relacional NoSQL (específicamente a *MongoDB*). Migrar de una base de datos relacional

a otra no relacional es bastante importante, pues en *MongoDB* el acceso a los datos es muy superior en lo referido al tiempo si lo comparamos con *MySQL*. Posteriormente, debo crear una aplicación web mediante un *Framework* para mostrar datos estadísticos en base a la información almacenada.

Una herramienta como esta permitiría por parte de la empresa que el tiempo dedicado al análisis de la información de los estudios radiológicos disminuyera, facilitando así la resolución de incidencias o cuestiones de cualquier índole, tanto informativa como analítica y predictiva.

La realización de este proyecto debe permitir extraer datos de diferentes orígenes, a saber: bases de datos relacionales, imágenes, informes médicos. Una vez realizado este primer paso, se debe migrar a una base de datos no relacional para finalmente crear una interfaz para mostrar todo tipo de informes estadísticos.

El alcance de esta aplicación radica en que servirá para enlazarla con futuros proyectos, ayudando a la toma de decisiones empresariales a la vez que será una herramienta útil para la extracción de modelos de predicción.

2.3. Planificación temporal de las tareas

La planificación de las tareas a realizar en la empresa son las siguientes:

- *Primera quincena:* Asimilación de conceptos relacionados con la empresa. Búsqueda de información e instalación de todo lo necesario. Estructuración esquemática de lo dispuesto a desarrollar.
- *Segunda quincena:* Primer contacto con la programación de la aplicación y corrección de todos los posibles errores que surjan.
- *Tercera y Cuarta quincena:* Realización de la aplicación y desarrollo web para la muestra de información.
- *Quinta y Sexta quincena:* Tratamiento con distintos tipos y cantidades de datos así como aplicar mejoras en la aplicación.

2.4. Recursos del proyecto

Para la realización del proyecto, Actualmed puso a mi disposición un ordenador con *Linux Mint* como sistema operativo. Además, me han creado un correo electrónico asociado a la

empresa con el que poder mantener el contacto con los trabajadores. Asimismo, con la utilización de dicho usuario también puedo acceder a un sistema de gestión de tareas para indicar en él mis tareas en curso, dudas, incidencias, etc.

En lo referente al software, tuve que instalar en dicho ordenador: Python, MongoDB, Flask, phpMyAdmin y librerías necesarias de Python.

2.5. Trabajo realizado en la empresa

Para iniciarme y contextualizar mi proyecto, instalé e immporté la Base de Datos **pacsdb** en mysql, además de instalar **phpMyAdmin** y **MongoDB**. Investigué por mi cuenta más a fondo el funcionamiento de MySQL, phpMyAdmin y MongoDB. Posteriormente, me informé sobre la programación en Python relacionada con las bases de datos, instalando así Pycharm que es un IDE (Integrated Development Environment) utilizado en programación con el que poder programar en Python. De esta forma, ahondé más en el autoaprendizaje del manejo de bases de datos tanto relacionales como no relacionales.

Me informé también sobre **DICOM**, aprendiendo que un archivo de estas características tiene dos partes: la *cabecera*, que es una tabla que incluye todos los datos de información del paciente y el *cuerpo* con los datos de la imagen. De esta manera, la información del paciente va siempre incluida en el mismo fichero que la imagen y la imagen va siempre perfectamente identificada. Por ese motivo instalé varios paquetes que necesitaba con los que poderme facilitar la muestra de aquellos datos más relevantes de la cabecera del DICOM utilizando Python. Ayudado tanto por compañeros de trabajo como por Internet, logré encontrar un módulo de Python llamado **pydicom**, ideal para tratar este tipo de archivos. Asimismo, también encontré otro paquete de Python llamado **MySQLdb**, con el cual pude manejar fácil e intuitivamente la base de datos de Actualmed (pacsdb) cuya programación se basa en MySQL.

Después de la etapa de aprendizaje y asimilación de conceptos, me dispuse a realizar mi aplicación. En ese instante me encontraba en la fase **ETL** (*Extract Transform Load*) de mi proyecto. Con tal de esclarecer mejor las ideas, decidí realizar cuatro programas que realizaran cada uno una tarea diferente para finalmente combinarlos y que realicen la tarea encomendada por ActualMed. Por este motivo, en ese momento tenía cuatro programas en desarrollo.

Debido al hecho de que la base de datos de la empresa (pacsdb) era harto compleja, decidí iniciarme en mis programas utilizando una parte mínima funcional de ella. Por ello, mi primer programa se basaba en la extracción de algunos datos en MySQL de un par de columnas de una de las tablas de pacsdb (en concreto, la de patient), el cual utilizaba la biblioteca MySQLdb. Mi segundo programa trataba archivos DICOM utilizando el módulo dicom de Python. A continuación, mi tercer programa manejaba la carga de datos en la base de datos no relacional

de MongoDB con el uso del módulo pymongo de Python. Finalmente, mi cuarto programa se centraba en la aplicación web. Esquemáticamente quedaría de esta forma:

- *Programa 1:* Extraer información relevante de la base de datos pacsdb de la empresa.
- *Programa 2:* Extraer información relevante de los archivos DICOM.
- *Programa 3:* Migrar información extraída a una base de datos NoSQL.
- *Programa 4:* Mostrar la información relevante mediante Flask.

En lo referido al ETL, los dos primeros programas se encargaban de la extracción (*Extract*) de la información significativa pertinente mientras que el tercero se ocupaba de la parte de carga (*Load*) de la información en una base de datos NoSQL o No relacional. En cuanto a la tarea de transformar toda esa información extraída en una estructura de datos (*Transform*), estaba incluida en los programas 1 y 2. En ellos trataba los datos de las tablas y los archivos DICOM conjuntamente, incluyendo toda la información extraída en un diccionario para tratar posteriormente cada una de las entradas del diccionario para extraer la información relevante y no repetida de los DICOM e introducirlos en este diccionario. Es decir:

1. Extraigo toda la información que se encuentra en las tablas, tanto datos como archivos DICOM, y los añado a un diccionario Python.
2. Obtengo una lista con todos los diccionarios, en el cual en cada uno de ellos se encuentra una imagen diferente con su correspondiente información.
3. Recorro cada uno de los diccionarios (es decir, cada una de las imágenes) y trato todos los archivos DICOM que contengan. Esto supone extraer todos los atributos posibles que contengan sus cabeceras sin tener en cuenta la imagen radiológica.
4. Esta información de la cabecera que he extraído de los DICOM las introduzco en el diccionario personal de la imagen, siempre y cuando esta información no se encuentre ya en el diccionario pues NO queremos información repetida.
5. Una vez hecho esto, borro el archivo binario DICOM del que he extraído los atributos, pues está escrito en binario (ininteligible en lenguaje humano) y ya no aporta nada útil.
6. Introduzco esta lista de diccionarios (que coincide con el número de imágenes) en una base de datos no relacional como lo es MongoDB.

Por último, utilicé el cuarto programa escrito en *Flask* para mostrar aquellos datos relevantes, a saber: número de estudios, número de series, número de instancias, número de bytes ocupados, etc. Cabe recordar que el programa está orientado a mostrar aquellos datos de interés que están almacenados en la base de datos no relacional (tarea realizada por los primeros programas).

2.6. Comparativas

Con el objetivo de obtener una aplicación más eficiente, realicé varias pruebas comparativas entre distintas formas de implementar y llevar a cabo la realización de lo comentado anteriormente.

Por un lado, implementé otro programa que hacía exactamente las mismas tareas que realizaba el anterior, pero esta vez utilizando exclusivamente el lenguaje de bases de datos relacional MySQL en lugar del lenguaje no relacional MongoDB. Todo ello con el propósito de comparar la eficiencia entre realizarlo de una forma o realizarlo de otra.

Utilizando cantidades de datos distintas, llegué a la conclusión de que si operaba utilizando poca cantidad de datos, el programa que utilizaba exclusivamente MySQL tardaba algo menos que el programa antiguo. A pesar de que parecía que no debería ser así, es normal que tardara algo menos, pues MongoDB funciona mejor en comparación con MySQL cuanta más cantidad de datos tiene, pues la forma en la que se almacenan es mucho más sencilla en Mongo. En lenguajes de bases de datos NoSQL se escalan y desnormalizan los datos.

Al comparar la velocidad de MongoDB con MySQL, los desarrolladores observan que esta última carece de velocidad y experimenta dificultades con grandes volúmenes de datos, por lo que sería una mejor opción para las empresas con bases de datos más pequeñas y que busquen una solución más general. Si bien esta es una de las ventajas de MongoDB sobre MySQL: la capacidad de hacer frente a grandes cantidades de datos no estructurados. [2]

Si ActualMed utilizara mi programa, les iría muchísimo más rápido el programa antiguo (el que utiliza MongoDB) que el programa de prueba con el cual estamos tratando ahora (el que utiliza exclusivamente MySQL). Esto se debería a la gran cantidad de datos que manejan ellos, aparte de la complejidad de los mismos, pues en la tabla de relaciones que me enseñaron el primer día donde se muestran todas las tablas que forman la base de datos de la empresa, pude observar la gran cantidad de relaciones complejas que tienen.

Por otro lado, también probé la posibilidad de añadir índices a mi programa para que fuera más rápido. Los índices de una base de datos es una estructura de datos que mejora la velocidad de las operaciones, por medio de identificador único de cada fila de una tabla, permitiendo un rápido acceso a los registros de una tabla en una base de datos. Similar al índice de un libro.

Tras varias pruebas con los mismos datos, llegué a la conclusión de que no había gran cambio en la eficiencia a la hora de mostrar los datos, pues los tiempos eran muy similares. Se podía deber al hecho explicado anteriormente: que no se tratan gran cantidad de datos reales y para lo que se utiliza actualmente no es el uso idóneo.

2.7. Resultados

Una vez realizados todos los programas comentados anteriormente, estructuré la jerarquía de carpetas de la forma en la que indica la Figura 2.1.

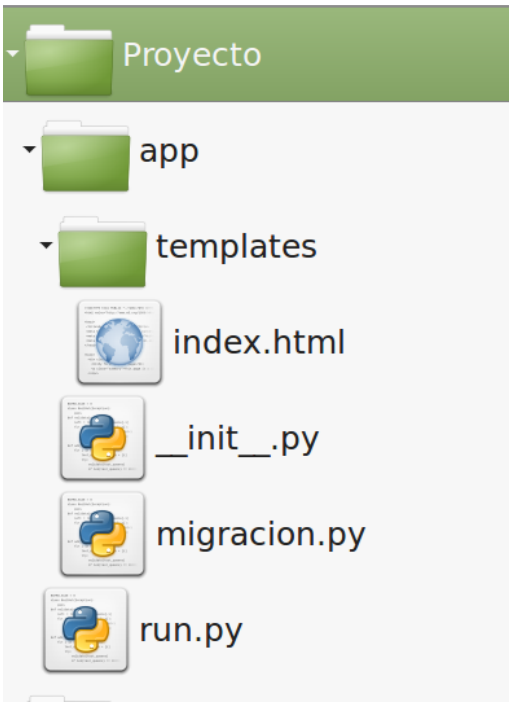


Figura 2.1: Jerarquía

El programa ***migracion.py*** que se encuentra en el Anexo A es el encargado de la migración de los datos de MySQL a MongoDB, anteriormente denotados como Programa 1, 2 y 3 del apartado anterior. El programa ***__init__.py*** del Anexo B es el que realiza las búsquedas necesarias en MongoDB para extraer la información que se mostrará en la aplicación web. El programa ***index.html*** del Anexo C se encuentra dentro de la carpeta *templates* ("plantillas" en español), pues precisamente está escrito en Flask y es el que lleva a cabo la estructuración en forma de tabla y posterior muestreo mediante web local de los datos recibidos del programa anterior. Estos dos últimos programas son los anteriormente denotados como Programa 4. Por último, el programa ***run.py*** únicmante ejecuta el programa principal *__init__.py* .

Una vez llegados al punto en el que los programas funcionaban, realicé el volcado de diversas bases de datos (las cuales ocupaban un total de 37 GB dadas en formato *.sql*) con el programa *migracion.py*. Dicho volcado tardó en realizarse dos días completos, pues la carga de trabajo a realizar era bastante considerable.

Una vez migrados todos los datos, ejecuté *run.py* desde consola como indica la imagen de la figura 2.2, la cual lanzaba la aplicación *__init__.py* encargada de extraer toda la información relevante de la base de datos de MongoDB. Además, esta hacía uso de la plantilla *index.html*, la cual lanzaba la aplicación web en local host.

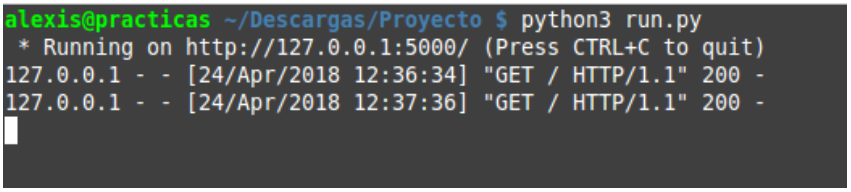


Figura 2.2: Ejecutando el programa desde consola

Abriendo el navegador e introduciendo la dirección mostrada en la imagen 2.2, nos presentaba por web local la tabla representada en la Figura 2.3.

Modalidad	Fecha inicio	Fecha fin	Días	Estudios	Estudios/Día	Mbytes/dia	Series	Instancias	Mbytes totales	Mbytes/estudio	Mbytes/serie	Mbytes/instancia	Instancia/serie
CR	2016-01-25 16:31:02	2018-01-22 13:52:01	727.89	8599.0	11.81	800578.46	10584.0	848.0	582733054.0	67767.54	55057.92	687185.21	55057.92
MR	2016-01-20 17:41:23	2017-12-27 18:58:08	707.05	17031557.0	24088.19	53460647.44	17460819.0	13324246.0	37799350772.0	2219.37	2164.81	2836.88	2164.81
PR	2018-01-22 13:48:15	2018-01-22 13:48:16	0.0	192.0	192.0	17280.0	360.0	472.0	17280.0	90.0	48.0	36.61	48.0
CT	2016-01-25 16:30:44	2017-12-09 05:19:01	683.53	2761567.0	4040.15	12409829.25	2965355.0	23430654.0	8482490584.0	3071.62	2860.53	362.03	2860.53
MG	2017-05-31 10:47:40	2017-05-31 10:48:38	0.0	636.0	636.0	131539962.0	663.0	24.0	131539962.0	206823.84	198401.15	5480831.75	198401.15
SC	2017-02-21 10:37:31	2017-12-08 06:49:41	289.84	79041.0	272.71	3423545.15	101433.0	186466.0	992280326.0	12554.0	9782.62	5321.51	9782.62
ECG	2017-11-03 15:19:17	2017-11-03 15:19:17	0.0	144.0	144.0	6606534.0	144.0	6.0	6606534.0	45878.71	45878.71	1101089.0	45878.71
OT	2017-11-03 10:30:35	2017-11-03 10:30:35	0.0	123.0	123.0	6607044.0	153.0	6.0	6607044.0	53715.8	43183.29	1101174.0	43183.29
PT	2016-01-25 16:31:15	2017-05-31 10:42:42	491.76	161049.0	327.5	212048.02	209487.0	1001622.0	104276732.0	647.48	497.77	104.11	497.77
XA	2016-03-01 13:08:28	2017-11-07 16:11:44	616.13	15600.0	25.32	175657.84	17225.0	7800.0	108228068.0	6937.7	6283.2	13875.39	6283.2
SR	2016-02-19 18:36:44	2017-12-26 18:11:04	675.98	42076.0	62.24	201150.24	29599.0	1360.0	135973542.0	3231.62	4593.86	99980.55	4593.86

Figura 2.3: Datos mostrados mediante la aplicación en Flask

Como podemos observar en la imagen 2.3, se ha realizado una agrupación por modalidades. Dentro de cada una de ellas, se ha obtenido las fechas de inicio y fin junto con los días transcurridos entre ambas fechas. Además, muestra la cantidad de estudios, series e instancias realizados, así como los MBytes ocupados. Por último, se establecen las relaciones entre estudios por día, instancias por series y MBytes por días, estudios, series e instancias.

2.8. Conclusiones

Desde el punto de vista personal, mi estancia en la empresa Actualmed me ha servido para mejorar en mis aptitudes como programador, así como para afrontar la realidad en cuanto al mundo empresarial, pues es algo a lo que un estudiante no suele estar acostumbrado y es algo importantísimo hoy en día.

Desde otro punto de vista más formal, mis objetivos en las prácticas se han llevado a cabo, pues he logrado realizar todo lo que se me propuso hacer. La planificación que me asigné ha sido realizada prácticamente conforme había sido descrita, realizando en menor o mayor medida todo lo expuesto.

A pesar de no haber obtenido una aplicación sumamente eficiente, pues siempre se puede mejorar en cualquier sentido, durante este corto trayecto he aprendido conceptos nuevos y al fin y al cabo, la aplicación realiza la función encomendada, que es lo verdaderamente importante en cuanto a mi estancia en Actualmed.

La herramienta que he logrado crear a la hora de mostrar aquellos datos que más le interesaba a la empresa serán utilizados por la misma como herramienta estadística de análisis, por lo que será de gran utilidad para el futuro de la empresa para alcanzar los objetivos que ella misma se proponga.

En definitiva, me quedo con que he realizado un buen trabajo y he aprendido todo lo que ha estado a mi alcance.

Capítulo 3

Estimación de curvatura en curvas discretas

3.1. Introducción

En matemáticas, la rama de la geometría se ocupa del estudio de las propiedades de las figuras en el plano, en el espacio, en $\mathbb{R}^4$, etc. En concreto, la **geometría diferencial** es el estudio de la geometría enfocada a variedades diferenciables.

Sin embargo, en la práctica, los experimentos solo pueden medir una cantidad finita de datos. De aquí surge la **Geometría Diferencial Discreta** (DDG en inglés por las siglas *Discrete Differential Geometry*), la cual se ocupa del estudio de las contrapartes discretas de la geometría diferencial aplicadas a entornos discretos.

El interés actual en DDG deriva no solo de su importancia en las matemáticas puras, sino también de sus aplicaciones en otros campos, incluyendo: Gráficos por Computadora (ejemplo en la Figura 3.1), Arquitectura, Física matemática, etc.

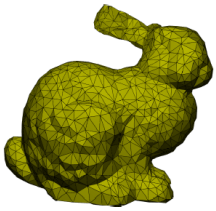


Figura 3.1: Aplicación DDG en gráficos por computadora

Objetivos

Aprovechando que la estancia en prácticas han sido la del tratamiento de imágenes de un centro radiológico, este trabajo final de grado se va a centrar en realizar una revisión de los conceptos básicos de las geometría diferencial para luego extrapolarlo al ámbito discreto. Dada la amplitud de la materia relacionada con DDG, este estudio se va a centrar en las nociones más básicas de esta, dando una serie de conceptos que en el entorno continuo son asequibles mientras que en el entorno discreto se complican.

Concretamente, este documento se va a centrar en realizar una estimación de la curvatura en curvas discretas. Para ello, el primer paso será el desarrollar la teoría sobre curvas discretas a partir de las definiciones básicas de las curvas diferenciables. Después, nos centraremos en una de las magnitudes más importantes en las curvas: la curvatura. Ahondaremos en la importancia de esta magnitud en las curvas diferenciables y en el entorno discreto.

3.2. Curvas planas diferenciables

Todos tenemos una idea intuitiva de lo que es una **curva**, en concreto una curva en el plano. Pero para trabajar con ella se necesita antes una definición matemática para así englobar la idea nacida con origen en nuestra intuición.

De esta forma, tenemos el concepto intuitivo de curva como conjunto continuo de puntos en el espacio o como trayectoria recorrida por un móvil.

Definición 1 *Dado un intervalo abierto I, se define una curva plana diferencial parametrizada como una aplicación diferenciable*

$$\alpha : I \to \mathbb{R}^2$$

la cual es al menos C^1, es decir, continua, derivable y con derivada continua en I. En términos matemáticos, una curva plana es una aplicación que tiene como imagen el subconjunto $\mathbb{R}^2$, parametrizando cada punto de este subconjunto por un número real.

La variable $t \in I$ hace referencia al **parámetro de la curva**, cuya interpretación física habitual es pensar que dicho parámetro representa el tiempo y que $\alpha(t)$ indica en qué posición del plano se encuentra una partícula en el instante t.

Definición 2 *Una curva diferenciable a trozos es una aplicación continua $\gamma : [a, b] \to \mathbb{R}^2$ tal que existen $s_0 = a < s_1 < \ldots < s_n = b$ de manera que para cualquier $i = 0, \ldots, n-1$, la aplicación $\gamma|_{[s_i, s_{i+1}]}$ es de clase C^1.*

Definición 3 *Un segmento de curva parametrizada es una aplicación $\alpha : [a, b] \to \mathbb{R}^2$ y un intervalo cerrado $[a, b]$ tal que existe una curva parametrizada $\beta :]c, d[\to \mathbb{R}^2$ tal que $[a, b] \subset]c, d[$ y $\beta(t) = \alpha(t)$ para todo $t \in [a, b]$*

Definición 4 *Una reparametrización de clase C^k con $k \geq 1$ es una biyección $g :]c, d[\to]a, b[$ tal que g y g^{-1} son de clase C^k. [3]*

Una vez dada una reparametrización, se puede definir otra curva $\beta :]c, d[\to \mathbb{R}^2$ como $\beta(r) = \alpha(g(r))$.

Ejemplos

A continuación se presentan una serie de ejemplos de parametrizaciones de curvas [3]:

Una *recta* viene representada por $\alpha : \mathbb{R} \to \mathbb{R}^2$ tal que $\alpha(t) = (x_0 + ta, y_0 + tb)$.

Una *circunferencia* viene dada por $\alpha : \mathbb{R} \to \mathbb{R}^2$ tal que $\alpha(t) = (x_0 + rcos(t), y_0 + rsen(t))$.

Una *elipse* viene dada por $\alpha : \mathbb{R} \to \mathbb{R}^2$ tal que $\alpha(t) = (x_0 + acos(t), y_0 + bsen(t))$.

Se conoce como *cicloide* a la curva descrita por un punto de la circunferencia de radio r, cuando esta rueda sin resbalar sobre una recta (Figura 3.2). Sus ecuaciones paramétricas vienen dadas por

$$\begin{cases} x(t) = r(t - sen(t)) \\ y(t) = r(1 - cos(t)) \end{cases}$$

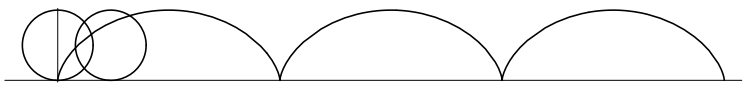


Figura 3.2: cicloide

3.2.1. Definiciones básicas

Definición 5 *El vector velocidad de una curva paramétrica diferenciable $\alpha : I \to \mathbb{R}^2$ en $t_0 \in I$ es el vector $\alpha'(t_0) = \frac{d\alpha}{dt}|_{t=t_0}$. La velocidad de α en $t = t_0$ es el módulo del vector velocidad.*

Si el vector velocidad en un punto no se anula, es decir $\alpha'(t_0) \neq 0$, podemos definir la recta que pasa por $\alpha(t_0)$ que tiene como vector director el vector velocidad. Esta recta es la *recta tangente* a la curva α en t_0. La existencia en cada punto de la curva de dicha recta es fundamental para el estudio de la geometría diferencial de la curva.

Definición 6 *Una curva plana regular es una curva paramétrica y diferenciable $\alpha : I \to \mathbb{R}^2$ tal que $\alpha'(t) \neq 0$ para cada $t \in I$.*

Definición 7 *El vector tangente a una curva regular α en un punto es el vector unitario que tiene la misma dirección que el vector velocidad, es decir,*

$$T(t) = \frac{\alpha'(t)}{||\alpha'(t)||}.$$

La recta tangente a un punto $\alpha(t_0)$ de una curva regular α es la recta que pasa por $\alpha(t_0)$ y que tiene como vector director el vector tangente $T(t_0)$, o equivalentemente, el vector velocidad $\alpha'(t_0)$.

Una de las primeras magnitudes que podemos asociar a una curva parametrizada es su longitud. Vamos a asociar cada segmento (ver Definición 3) de la curva regular, definida en un intervalo cerrado $\alpha : [a, b] \to \mathbb{R}^2$, un valor real, a saber: $l(\alpha)$. La longitud de un segmento de recta es una cantidad que podemos definir sin ningún problema. Si $\alpha(t) = (x_1 + tv_1, x_2 + tv_2)$, entonces su longitud es

$$l(\alpha) = ||bv_1 - av_1, bv_2 - av_2)||.$$

Es decir, la longitud de un segmento de recta es la distancia entre el punto inicial y el final del segmento.

Una curva poligonal es una curva plana diferenciable a trozos tal que las partes que sí que son diferenciable son segmentos de recta. La longitud de una curva poligonal es la suma de las longitudes de los segmentos que la forman.

Definición 8 *Una partición del intervalo $[a, b]$ es un subconjunto finito y ordenado de la forma $\mathcal{P} = \{a = t_0 < t_1 < \cdots < t_n = b\}$. Para cada partición del intervalo $[a, b]$ se puede definir la poligonal asociada $\mathcal{P}_\alpha$ que tiene como vértices los puntos $\alpha(t_i)$ para $i = 0, \cdots, n$.*

La longitud de la poligonal es: $l(\mathcal{P}_\alpha) = \sum_{i=1}^{n} ||\alpha(t_{i-1}) - \alpha(t_i)||$. La norma de una partición $\mathcal{P}$ se define como $N(\mathcal{P}) = \max_{i=1,\cdots,n} |t_i - t_{i-1}|$.

Definición 9 *La longitud de la curva α se define como el límite de las longitudes de las poligonales asociadas cuando la norma de la poligonal tiende a 0. Es decir*

$$l(\alpha) = \lim_{N(\mathcal{P})\to 0} l(\mathcal{P}_\alpha).$$

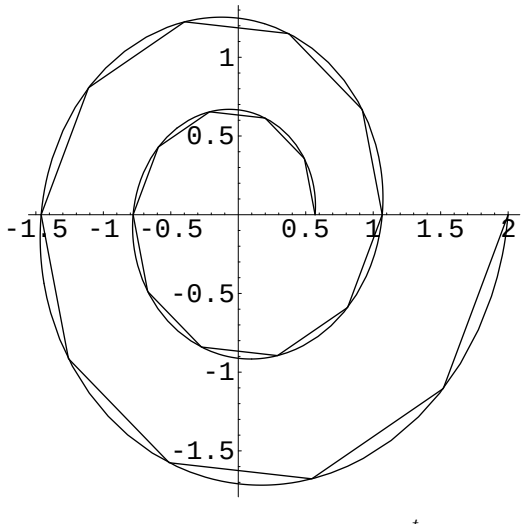


Figura 3.3: Una poligonal asociada a una curva

Si $\beta :]c, d[\to \mathbb{R}^2$ es una curva parametrizada, la restricción de β a un intervalo $[a, b] \subset]c, d[$ y que denotaremos por $\beta_{[a,b]}$ es un segmento de curva parametrizada. La longitud de la curva β desde $\beta(a)$ hasta $\beta(b)$ es la longitud del segmento $l(\beta_{[a,b]})$.

Cabe destacar que la clase de curvas estudiadas aquí son curvas regulares a trozos, por lo que el hecho de que sean diferenciables nos asegura la existencia del límite.

Proposición 1 *La longitud de un segmento de curva parametrizada $\alpha : [a, b] \to \mathbb{R}^2$ se define como*

$$l(\alpha) = \int_a^b ||\alpha'(t)||dt.$$

Definición 10 *Diremos que una curva parametrizada está parametrizada por longitud de arco (PLA) si $||\alpha'(t)|| = 1$ para todo $t \in I$.*

Por lo general, denotamos el parámetro de longitud de arco de una curva por s.

Cabe destacar que si una curva está parametrizada por longitud de arco, entonces es una curva regular. Este parámetro es interesante desde el punto de vista operacional porque simplifica los cálculos y desde el punto de vista físico porque la curva representa un móvil de velocidad constante igual a la unidad. [3]

Si $\alpha : [a, b] \to \mathbb{R}^2$ es un segmento de curva regular PLA, entonces la longitud del segmento de curva definido en el intervalo $[a, s] \subset [a, b]$ es $l(\alpha_{[a,s]}) = s - a$. En el caso de que $a = 0$, entonces $l(\alpha_{[a,s]}) = s$, de ahí el nombre de parámetro de longitud de arco.

Proposición 2 *Toda curva regular $\alpha :]a, b[\to \mathbb{R}^2$ puede ser reparametrizada (ver Definición 4) por longitud de arco con un cambio de parámetro que conserva la orientación. [3]*

Consideramos una curva regular PLA $\alpha : I \to \mathbb{R}^2$ y una orientación ***or*** del espacio vectorial $\mathbb{R}^2$. Por tanto, para cada $s \in I$ sabemos que existe un vector unitario $N(s)$ (llamado vector normal) ortogonal a T(s) de tal forma que la base $\{T(s), N(s)\}$ define una base del espacio vectorial $\mathbb{R}^2$ adaptada a la curva que definen la misma orientación que *or*. [3]

Definición 11 *El vector normal a una curva regular α en un punto es el vector unitario perpendicular al vector tangente.*

La recta normal a un punto $\alpha(s)$ es la recta que pasa por $\alpha(s)$ y que tiene como vector director el vector normal $N(s)$, o equivalentemente, el vector ortogonal a $T(s)$.

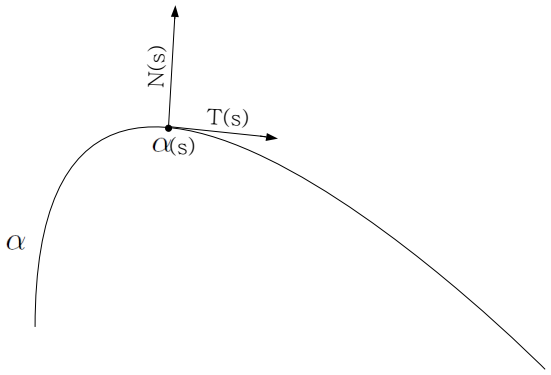


Figura 3.4: Representación del vector tangente y normal

3.2.2. Curvatura en curvas planas diferenciables

Partimos de una curva plana diferenciable $\alpha : I \to \mathbb{R}^2$. Una de las magnitudes que podemos asociar a una curva plana es la llamada **curvatura** κ.

En concreto, la curvatura de una curva α en un punto mide la desviación de la curva respecto de la recta tangente que pasa por el mismo punto, es decir, es una medida del cambio de dirección del vector tangente a una curva. [3]

Definición 12 *Sea α una curva parametrizada por longitud de arco (PLA). Tomamos $s \in I$ como el parámetro para dicha curva. Entonces, la curvatura es la magnitud de la tasa de variación del vector tangente T (el cual nos da la velocidad del vector) interpretado como*

$$\kappa(s) = ||T'(s)|| = ||\alpha''(s)||.$$

*De esta forma, esta magnitud representa la aceleración de la partícula. Cabe destacar que la curvatura es un término **local**, es decir, que $\kappa(s)$ solo depende en α del entorno directo de s.*

Proposición 3 *Sea α una curva dada por $\alpha(t) = (x(t), y(t))$, entonces se cumple que [3]*

$$\kappa(t) = \frac{x'y'' - x''y'}{(x'^2 + y'^2)^{\frac{3}{2}}}(t).$$

Proposición 4 *La curvatura de una línea recta es siempre nula, es decir, $\kappa = 0$.*

Proposición 5 *La curvatura de una circunferencia de radio R es igual al recíproco de su radio, es decir, $\kappa = \frac{1}{R}$. Se ve claramente como la curvatura deberá ser más grande (más pronunciada) cuanto menor sea R (círculos más pequeños) y deberá ser más pequeña (menos pronunciada) cuando mayor sea R (círculos más grandes).*

Curvatura plana a partir de círculos osculadores

En realidad, años antes de la definición de curvatura dada en la Definición 12, *Leibniz* planteó la definición de curvatura partiendo de otros conceptos. Este matemático dedujo las Proposiciones 4 y 5 de forma intuitiva y buscó una definición de curvatura que reflejara el comportamiento de la curvatura de una circunferencia frente a su radio. Su aproximación hizo uso del llamado *círculo osculador* de una curva en un punto, el cual se define como la circunferencia que mejor aproxima a la curva en dicho punto. [4]

Sea α una curva plana diferenciable. Dado un punto $p = \alpha(s)$, existen muchos círculos tangentes a α en p, es decir, todos aquellos círculos cuya velocidad en p sea la misma que la

de α, o equivalentemente, aquellos círculos cuyo centro se encuentre en la recta que pasa por p y es ortogonal al vector tangente $T(s)$. Entre todos esos círculos hay exactamente uno cuya aceleración en p es la misma que la de α, o lo que es lo mismo, existe un único círculo el cual mejor aproxima a la curva cerca de p. Dicho círculo es conocido como el ***círculo osculador***. De todos los círculos tangentes a la curva en la Figura 3.5, solamente sería osculador el más sombreado.

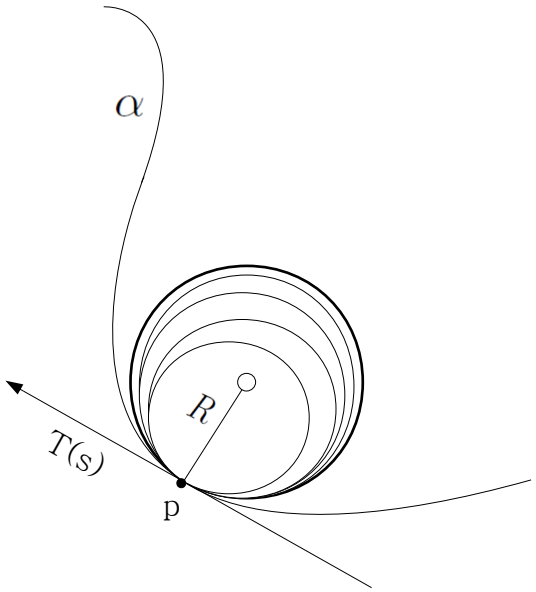


Figura 3.5: Círculo osculador en una curva plana diferenciable

Definición 13 *Sea α una curva plana regular definida en I y sea $s \in I$ tal que $\kappa(s) \neq 0$. Se define el círculo osculador a la curva α en el punto $p = \alpha(s)$ como la circunferencia de radio $\frac{1}{|\kappa(s)|}$ y de centro $\alpha(s) + \frac{1}{\kappa(s)}N(s)$.*

El centro y el radio del círculo osculador en un punto de la curva son llamados centro de curvatura y radio de curvatura de la curva en ese punto.

La importancia de esta definición reside en que el círculo osculador a una curva en un punto dado es una circunferencia cuyo centro se encuentra sobre la recta normal a la curva y tiene la misma curvatura que la curva dada en ese punto.

Proposición 6 *La curvatura de α en $p = \alpha(s)$ se define como la curvatura de su círculo osculador, es decir, como el inverso del radio de curvatura $\kappa = \frac{1}{R}$. [4]*

3.3. Curvas planas diferenciables discretas

A continuación desarrollaremos la teoría sobre curvas geométricas discretas de forma semejante a como se ha hecho con curvas diferenciales. Para ello, se necesita definir una curva discreta, la cual tomaremos como una secuencia de puntos en el plano conectados mediante líneas rectas.

Definición 14 *Dado un intervalo I, se define una* ***curva plana discreta*** *en $\mathbb{R}^2$ como una curva diferenciable a trozos donde cada trozo es un segmento entre sus vértices. Dichos vértices están definidos como una tupla ordenada $(\gamma_0, \gamma_1, \cdots, \gamma_{n-1}) \in \mathbb{R}^{2n}$ donde los* ***vértices*** *$\gamma_i \in \mathbb{R}^2$ con $\gamma_{i+1} \neq \gamma_i$ para todo i.*

Como se puede observar en la Figura 3.6, nos encontramos ante un ejemplo de una curva discreta γ, la cual es una función que toma cada uno de los puntos del intervalo $I = [0, L]$ de la recta real.

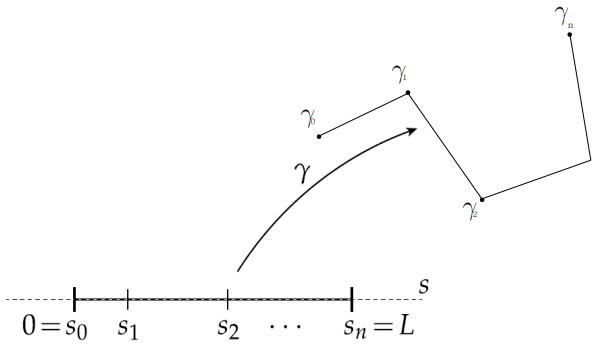


Figura 3.6: Ejemplo de una curva discreta en el plano

3.3.1. Definiciones básicas

Definición 15 *La curva plana diferenciable discreta γ es una curva regular si para cualquiera tres puntos sucesivos $\gamma_{i-1}, \gamma_i, \gamma_{i+1}$ son diferentes para todos los $i-1, i, i+1 \in I$.*

Definición 16 *Entre cada par consecutivo de vértices $(i, i+1)$ donde $i, i+1 \in I$, se define el vector de aristas*

$$E_i = \gamma_{i+1} - \gamma_i.$$

Si para cada pareja de puntos de γ se cumple que $||E_i|| > 0$, es decir, que son diferentes, entonces el vector de aristas nos permite definir el **vector tangente** unitario a lo largo de cada arista.

Definición 17 *Se define el vector tangente unitario entre cada par consecutivo de vértices* $(i, i+1)$ *donde* $i, i+1 \in I$ *como*

$$T_i = \frac{E_i}{||E_i||} = \frac{\gamma_{i+1} - \gamma_i}{||\gamma_{i+1} - \gamma_i||}.$$

Tanto los vectores de aristas E_i como los vectores tangentes T_i se definen en las aristas en lugar de definirse en los vértices a pesar de la notación.

Definición 18 *[5] Se define la longitud de arco total de la curva* γ *como*

$$L(\gamma) = \sum_{i,i+1 \in I} ||\gamma_{i+1} - \gamma_i|| = \sum_{i,i+1 \in I} ||E_i||.$$

Además, γ *estará parametrizada por longitud de arco (PLA) si* $||E_i|| = 1$ *para todo* $i \in I$.

De esta forma, para una curva discreta PLA γ, el vector tangente y el vector arista coinciden $T_i = E_i = \gamma_{i+1} - \gamma_i$.

En la Figura 3.7 se pude ver representada una parte de una curva discreta PLA cuyos vértices cualesquiera son γ_{i-1}, γ_i y γ_{i+1}. Junto con ellos se encuentran las aristas asociadas a dichos vértices E_{i-1} y E_i.

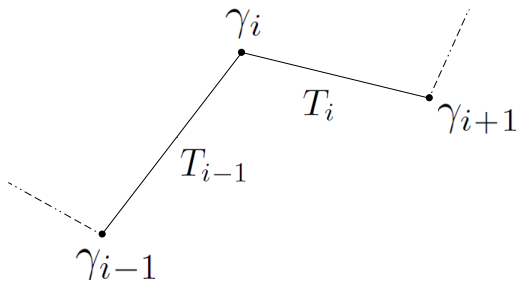


Figura 3.7: Partes de una curva discreta PLA

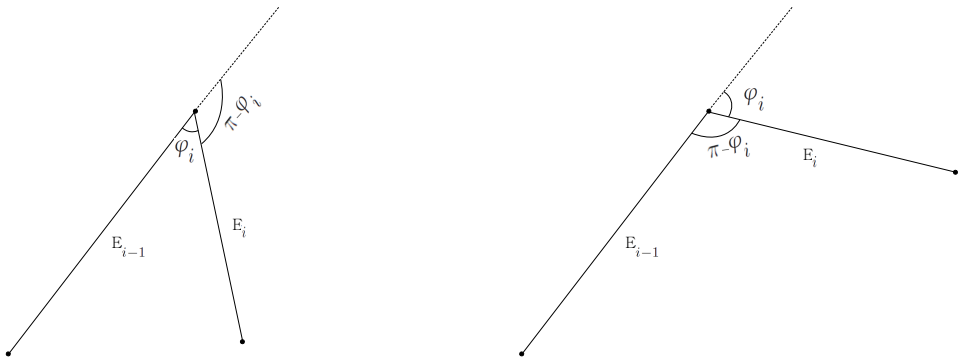
Cuando dos rectas secantes se cortan, determinan cuatro ángulos iguales dos a dos (uno grande y otro pequeño). De esta forma, podemos considerar el ángulo que forman dos aristas.

Definición 19 *Dada* $\gamma : I \to \mathbb{R}^2$ *una curva discreta, se define*

$$\varphi_i = \angle(E_i, E_{i-1}) = \angle(T_i, T_{i-1}) \in [-\pi, \pi]$$

como el ángulo más pequeño formado entre las aristas E_i *y* E_{i-1}*, o lo que es lo mismo, entre sus rectas tangentes* T_i *y* T_{i-1}*. El ángulo grande vendría dado por* $(\pi - \varphi_i) \in [-\pi, \pi]$*.*

En la Figura 3.8 vemos dos ejemplos del ángulo de una curva discreta.



(a) (b)

Figura 3.8: Ángulo de giro entre dos vértices de una curva discreta

Ejemplos

A continuación se presentan una serie de ejemplos de parametrizaciones de curvas [6]:

Un *polígono regular* de n lados inscrito en un círculo de radio $R = \frac{1}{2sen\frac{\pi}{n}}$ se parametriza por longitud de arco. Por supuesto, puede verse como un círculo discretizado.

Se conoce como *cicloide discreta* a la curva descrita por un vértice de un polígono regular de n lados cuando este gira sobre una recta. La curva discreta resultante viene dada por

$$\gamma_i = \sum_{j=0}^{i} \left(1 - e^{-Ij\frac{2\pi}{n}}\right)$$

siendo $I = \sqrt{-1}$ la unidad imaginaria.

En la Figura 3.9 se observa la cicloide en la que gira un octógono.

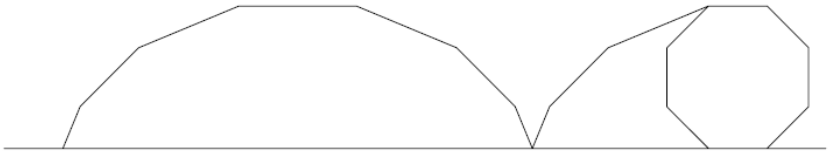


Figura 3.9: Cicloide discreto

3.3.2. Curvatura en curvas discretas

Una de las grandes incógnitas que uno se plantea cuando define las curvas discretas es cómo se podría determinar su **curvatura**. La primera definición que se ha propuesto para las curvas planas diferenciales ha sido el describir la curvatura como la tasa de variación del vector tangente, es decir, definir la curvatura como la **segunda derivada** $\kappa(s) = ||\alpha''(s)||$. Si estableciéramos la misma idea para las curvas discretas, nos encontraríamos con el grave problema de que en las líneas rectas la curvatura valdría cero mientras que en los vértices quedaría indefinida.

Es por ello que vamos a tomar de la definición de curvatura en curvas diferenciables a partir de **círculos osculadores**. Como se ha comentado anteriormente (ver Proposiciones 5 y 6), los círculos osculadores tienen la misma curvatura que la curva dada en un punto. Es importante recordar que la curvatura del círculo osculador se define como el inverso del radio de curvatura.

Debido a que la curvatura es un término local, la **curvatura discreta** κ_i va a depender de la longitud de las aristas y de los ángulos cercanos a γ_i, es decir, de $||E_i||$, $||E_{i-1}||$ y φ_i.

Una vez llegados a este punto consideraremos tres posibilidades de definir dichos círculos en el entorno discreto para así obtener diferentes nociones de curvatura discreta: a partir del vértice, a partir de la arista y a partir de curvas PLA. La explicación de cada una de ellas y las comparativas se irán viendo a lo largo de esta sección y la siguiente.

Círculo osculador a partir del vértice

El círculo osculador a partir del *vértice* γ_i es el círculo que pasa por γ_i y sus dos vecinos γ_{i-1} y γ_{i+1}. Su centro viene dado por la intersección de las dos mediatrices de las aristas adyacentes E_i y E_{i-1}. Véase Figura 3.10.

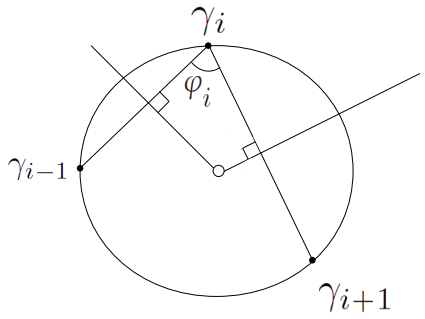


Figura 3.10: Círculo osculador a partir del vértice

Para obtener la primera definición de curvatura, procedemos a enunciar un teorema elemental de la trigonometría:

Teorema 1 *[7] Teorema de los senos. Sea un triángulo con vértices ABC donde a, b, c son los lados opuestos a los ángulos α, β, γ respectivamente. Además, R denota el radio de la circunferencia circunscrita (ilustrado en la Figura 3.11). Entonces* $\frac{a}{sen\alpha} = \frac{b}{sen\beta} = \frac{c}{sen\gamma} = 2R$.

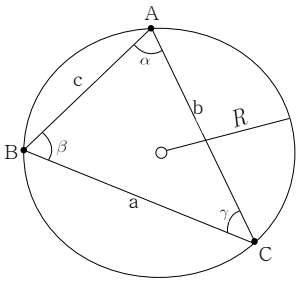


Figura 3.11: Teorema de los senos

Si formáramos un triángulo de vértices γ_{i-1}, γ_i y γ_{i+1}, obtendríamos la Figura 3.12. Aplicando el *Teorema de los senos* 1 a dicha Figura, se deduce que $\frac{||\gamma_{i+1}-\gamma_{i-1}||}{sin\varphi_i} = 2R$. Ahora que conocemos el radio del círculo osculador a partir del vértice, vamos a deducir la curvatura que surge a partir de él.

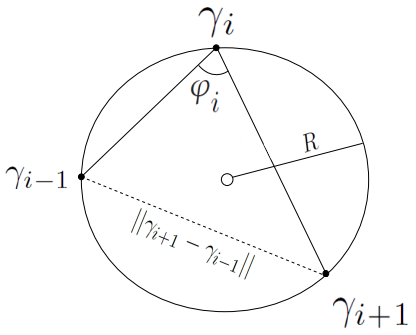


Figura 3.12: Teorema de los senos aplicado al círculo osculador a partir del vértice

Proposición 7 *Sabemos que el radio del círculo osculador a partir del vértice viene dado por* $R = \frac{||\gamma_{i+1} - \gamma_{i-1}||}{2sin\varphi_i}$. *De esta forma se obtiene la curvatura*

$$\kappa_i = \frac{1}{R} = \frac{2sin\varphi_i}{||\gamma_{i+1} - \gamma_{i-1}||}.$$

Círculo osculador a partir de la arista

El círculo osculador a partir de la *arista* E_i es el círculo tangente a las tres aristas sucesivas E_{i-1}, E_i y E_{i+1}. Su centro viene dado por la intersección de las dos bisectrices de los ángulos φ_i y φ_{i+1}. En la Figura 3.13 se puede ver representado este esquema acabado de explicar.

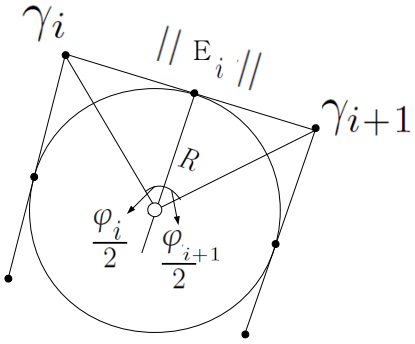


Figura 3.13: Círculo osculador a partir de la arista

Cabe destacar que el círculo osculador a partir de la arista es determinado de forma única siempre que las tres aristas consecutivas tengan direcciones diferentes, incluso aunque la curva no sea convexa. Un claro ejemplo ilustrativo se encuentra en la Figura 3.14.

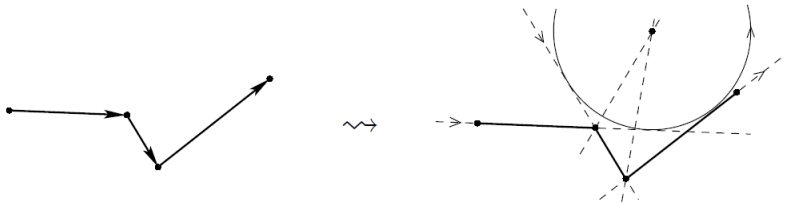


Figura 3.14: Círculo osculador a partir de la arista de una curva discreta no convexa

El radio de curvatura del círculo osculador viene dado por $||E_i|| = R(tan\frac{\varphi_i}{2} + tan\frac{\varphi_{i+1}}{2})$. [5]

Proposición 8 *Sabemos que el radio del círculo osculador a partir de la arista viene dado por* $R = \frac{||E_i||}{tan\frac{\varphi_i}{2}+tan\frac{\varphi_{i+1}}{2}}$. *De esta forma se obtiene la curvatura*

$$\kappa_i = \frac{1}{R} = \frac{tan\frac{\varphi_i}{2} + tan\frac{\varphi_{i+1}}{2}}{||E_i||}.$$

Círculo osculador a partir de curvas parametrizadas por longitud de arco (PLA)

Dada una curva discreta parametrizada por longitud de arco γ, se define el círculo osculador a partir de *curvas PLA* en el vértice γ_i como el círculo que toca las dos aristas E_{i-1} y E_i en sus puntos medios. Debido a que estamos ante una curva PLA, también podríamos considerar los vectores tangentes T_{i-1} y T_i en lugar de las aristas, pues son iguales.

Su centro viene dado por la intersección de las dos líneas divisorias de las dos aristas. Cabe destacar que como estamos ante una curva discreta PLA, la longitud de sus aristas será $||E_i|| = 1$. Como las aristas se dividen por sus puntos medios, cada lado medirá $\frac{1}{2}$. En la Figura 3.15 se puede ver representado este esquema acabado de explicar.

Para obtener la definición de curvatura partiendo de este círculo osculador, cabe recordar que en trigonometría, la *tangente* de un ángulo en un triángulo rectángulo se define como la razón entre el cateto opuesto al ángulo y el cateto adyacente. De esta forma se deduce que $tan\frac{\varphi_i}{2} = \frac{1/2}{R}$.

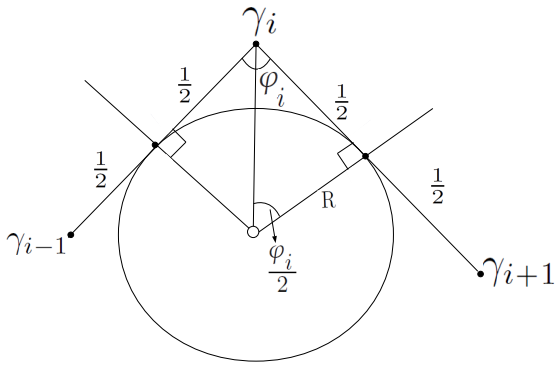


Figura 3.15: Círculo osculador a partir de curvas PLA

Proposición 9 *Sabemos que el radio del círculo osculador a partir de curvas PLA viene dado por* $R = \frac{1}{2tan\frac{\varphi_i}{2}}$. *De esta forma se obtiene la curvatura*

$$\kappa_i = \frac{1}{R} = 2tan\frac{\varphi_i}{2}.$$

Cabe destacar que dicha curvatura dará 0 en vértices rectos mientras que tenderá a ∞ en vértices no regulares.

NOTA. ***Aplicable a cada noción de curvatura discreta***

Como se ha comentado en la Definición 19 de ángulo de una curva discreta, φ_i es el ángulo más pequeño. En el caso de que quisiéramos considerar el ángulo grande en lugar del pequeño por conveniencia, comodidad o porque simplemente es el único ángulo que conocemos, el razonamiento anterior sería análogo pero en lugar de considerar φ_i sustituiríamos las deducciones por $\pi - \varphi_i$.

3.4. Comparaciones de las curvaturas

La definición de hasta tres curvaturas diferentes para curvas planas discretas nos hace preguntarnos cuál es mejor. Debido al hecho de que cada una de ellas cumple unas propiedades distintas e incluso su uso se restringe a curvas con ciertas características, dependerá del problema a resolver.

En el Cuadro 3.1 se muestran cada una de las curvaturas obtenidas.

Círculo osculador	**Curvatura** (Referencia)
a partir del vértice	$\kappa_i = \frac{2sin\varphi_i}{\|\|\gamma_{i+1}-\gamma_{i-1}\|\|}$ (Prop. 7)
a partir de la arista	$\kappa_i = \frac{tan\frac{\varphi_i}{2}+tan\frac{\varphi_{i+1}}{2}}{\|\|\gamma_{i+1}-\gamma_i\|\|}$ (Prop. 8)
a partir de curvas PLA	$\kappa_i = 2tan\frac{\varphi_i}{2}$ (Prop. 9)

Cuadro 3.1: Diferentes definiciones de curvaturas discretas

Para un análisis más amplio, vamos a compararlas mediante varios ejemplos. Para ello vamos a hacer uso de la programación para poderlo comparar con más eficiencia, precisión y variabilidad.

Comenzaremos con un ejemplo sencillo utilizando un polígono regular y a continuación seguiremos comparando las curvaturas con un polígono irregular, siendo ambas curvas discretas.

3.4.1. Comparación con polígono regular

El programa ***ComparacionCurvaturas.m*** que se encuentra en el Anexo D es un programa en Matlab que nos permitirá comparar las curvaturas mediante la definición de un polígono regular.

El programa lo que hace es definir una circunferencia y un polígono regular inscrito en ella. Se podrían definir con cualquier radio y cualquier número de lados. Para hacerse una idea, se puede ver en la Figura 3.16 la salida gráfica del programa para una circunferencia de radio 2 con un hexágono inscrito en ella.

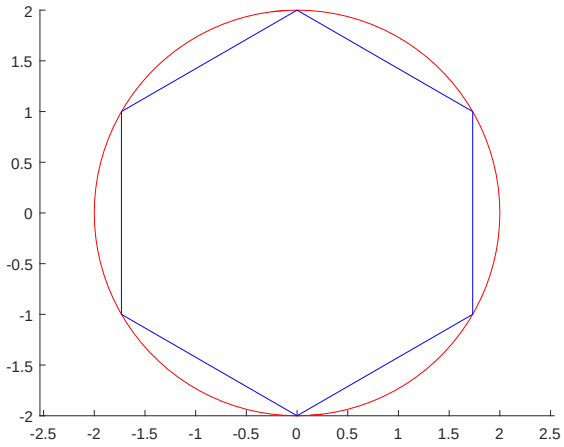


Figura 3.16: Salida gráfica del programa ComparacionCurvaturas.m

A continuación, lo que hace el programa es determinar el ángulo del polígono. Como el polígono de n lados es regular, cumple que todos sus ángulos miden $ang\_rad = \frac{(n-2)*180}{n}$ grados, que pasado a radianes es $ang = \pi - \frac{ang\_rad*\pi}{180}$. Seguidamente, calcula la curvatura de la circunferencia de radio R que viene dada por $\kappa = \frac{1}{R}$.

Como esta circunferencia actúa de círculo osculador para el polígono regular, todas las curvaturas de los vértices de dicho polígono deberían aproximarse lo máximo posible a la curvatura de la circunferencia.

Seguidamente, procede a calcular cada una de las curvaturas en cada vértice del polígono mediante la utilización de las curvaturas del Cuadro 3.1. Como es un polígono regular, es decir, que todos sus lados y ángulos interiores son iguales entre sí, las curvaturas dadas en cada vértice deberían dar el mismo valor.

Para el ejemplo concreto de la Figura 3.16 se obtiene la salida siguiente:

Curvatura del círculo: ***0.5***
Curvatura a partir de los vértices: ***0.5***
Curvatura a partir de las aristas: ***0.57735***
*Curvatura a partir de curvas PLA: 1.****1547***

Como la circunferencia es de radio 2, su curvatura es $\kappa = \frac{1}{2} = 0{,}5$. Podemos observar como la *curvatura a partir de los vértices* es exactamente igual que a la curvatura de su círculo osculador,

por lo que estaría perfectamente aproximada a la realidad. En cuanto a la *curvatura a partir de las aristas* observamos como se aproxima bastante a 0,5. En cambio, la *curvatura a partir de curvas PLA* se aleja mucho de la realidad, pues se diferencia en más de 0,6. Probemos ahora con un polígono regular de 3 lados, es decir, un triángulo. La salida por pantalla del programa es la siguiente:

Curvatura del círculo: ***0.5***
Curvatura a partir de los vértices: ***0.5***
Curvatura a partir de las aristas: ***1***
Curvatura a partir de curvas PLA: ***3.4641***

Vemos como la *curvatura a partir de los vértices* es exactamente la misma que la curvatura del círculo. Por otro lado, tanto la *curvatura a partir de las aristas* como la *curvatura a partir de curvas PLA* se alejan mucho más que antes del valor real. Probando con un polígono regular de 12 lados, es decir, un dodecágono, la salida por pantalla del programa es la siguiente:

Curvatura del círculo: ***0.5***
Curvatura a partir de los vértices: ***0.5***
Curvatura a partir de las aristas: ***0.51764***
Curvatura a partir de curvas PLA: ***0.5359***

Al aumentar el número de lados del polígono esclarecemos que las *curvatura a partir de las aristas* se aproxima más al valor real, al igual que sucede con la *curvatura a partir de curvas PLA*. La salida para un polígono de 200 lados es:

Curvatura del círculo: ***0.5***
Curvatura a partir de los vértices: ***0.5***
Curvatura a partir de las aristas: ***0.50006***
Curvatura a partir de curvas PLA: ***0.031419***

Dicha salida nos indica que cuan mayor es el número de lados del polígono, es decir, a medida que dicho polígono empieza a parecerse más a una circunferencia, las curvaturas a partir de los vértices y a partir de las aristas se asemejan e incluso llegan al número exacto de la curvatura real, a saber: la de su círculo osculador. En cambio, la curvatura a partir de curvas PLA se aleja cada vez más de su valor real, debido al hecho de que el polígono que estamos tratando no está parametrizado por longitud de arco, por lo que la fórmula (basada en ese principio) pierde todo el sentido. Concretamente para nuestro ejemplo, si dividiéramos la expresión obtenida de

curvatura a partir de curvas PLA por la norma de los vértices, es decir, $\kappa i\_pla = \frac{2tan(\frac{ang}{2})}{norm(v\_pos - v\_act)}$, estaríamos obteniendo exactamente la fórmula de la curvatura a partir de las aristas.

Cabe destacar que el radio del círculo osculador no influye en las conclusiones obtenidas en la comparación acabada de realizar, pues el razonamiento sería análogo.

3.4.2. Comparación con polígono irregular

El programa ***ComparacionCurvaturasIrregular.m*** que se encuentra en el Anexo E es un programa en Matlab que nos permitirá comparar las curvaturas mediante la definición de un polígono irregular.

El programa lo que hace es definir una circunferencia de radio 5 y un polígono irregular. Para hacerse una idea, se puede ver en la Figura 3.17 la salida gráfica del programa a la que se ha añadido posteriormente los nombres de los vértices y algunas aristas para más claridad.

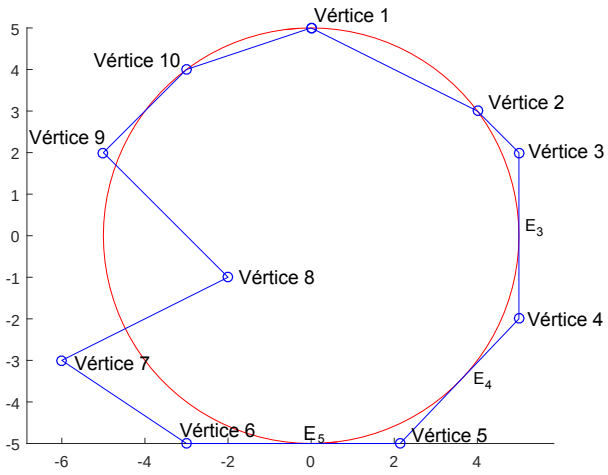


Figura 3.17: Salida gráfica del programa ComparacionCurvaturasIrregular.m

A continuación, lo que hace el programa es determinar los vectores arista E_i a partir de la Definición 16. Una vez los tiene calculados, determina el ángulo φ_i que forman cada una de estas aristas. Lo hace mediante la definición del ángulo que forman dos rectas con vectores directores $\vec{u}$, $\vec{v}$ respectivamente: $cos(\alpha) = \frac{|u_1 \cdot v_1 + u_2 \cdot v_2|}{\sqrt{u_1^2 + u_2^2}\sqrt{v_1^2 + v_2^2}}$. [8]

Seguidamente, calcula la curvatura de la circunferencia de radio 5 dada por $\kappa = \frac{1}{5} = 0{,}2$. Más adelante, procede a calcular cada una de las curvaturas en cada vértice del polígono irregular mediante la utilización de las curvaturas del Cuadro 3.1.

Se obtiene la salida siguiente:

Curvatura del círculo: ***0.2***

Vértice 1:
Curvatura a partir de los vértices: ***0.2***
Curvatura a partir de las aristas: ***0.18524***
Curvatura a partir de curvas PLA: ***0.82843***

Vértice 2:
Curvatura a partir de los vértices: ***0.10847***
Curvatura a partir de las aristas: ***0.2295***
Curvatura a partir de curvas PLA: ***0.32456***

Vértice 3:
Curvatura a partir de los vértices: ***0.27735***
Curvatura a partir de las aristas: ***0.20711***
Curvatura a partir de curvas PLA: ***0.82843***

Vértice 4:
Curvatura a partir de los vértices: ***0.1825***
Curvatura a partir de las aristas: ***0.19315***
Curvatura a partir de curvas PLA: ***0.8005***

Vértice 5:
Curvatura a partir de los vértices: ***0.16943***
Curvatura a partir de las aristas: ***0.16664***
Curvatura a partir de curvas PLA: ***0.85655***

Vértice 6:
Curvatura a partir de los vértices: ***0.13235***
Curvatura a partir de las aristas: ***0.16795***
Curvatura a partir de curvas PLA: ***0.60555***

Vértice 7:
Curvatura a partir de los vértices: ***0.42116***
Curvatura a partir de las aristas: ***0.25953***
Curvatura a partir de curvas PLA: ***1.1606***

Vértice 8:
Curvatura a partir de los vértices: ***0.3721***
Curvatura a partir de las aristas: ***0.33977***
Curvatura a partir de curvas PLA: ***1.4415***

Vértice 9:
Curvatura a partir de los vértices: ***0.39223***
Curvatura a partir de las aristas: ***0.70711***
Curvatura a partir de curvas PLA: ***2***

Vértice 10:
Curvatura a partir de los vértices: ***0.15339***
Curvatura a partir de las aristas: ***0.1493***
Curvatura a partir de curvas PLA: ***0.47214***

En cuanto a las curvaturas *a partir de curvas PLA* dadas en la salida, reparamos como nos dan valores muy variados. Esto se debe a que el polígono irregular que acabamos de crear no está Parametrizado por Longitud de Arco y por tanto no tiene ningún sentido calcular su curvatura a partir de esta definición.

Centrándonos en el *Vértice 1*, observamos como el círculo rojo de la Figura 3.17 actúa como círculo osculador a partir del vértice 1, pues pasa por el vértice 1 y sus dos vecinos (vértices 10 y 2). De esta forma, sabemos con certeza que su curvatura debe dar 0,2, ya que es la misma que la de su círculo osculador. En efecto, vemos en la salida anterior como la curvatura dada por la fórmula del círculo osculador a partir del vértice es exactamente dicho valor. Además, la curvatura dada por la fórmula del círculo osculador a partir de la arista da 0,18, un valor muy próximo al valor real.

Centrándonos en los *Vértices 3* y *4*, nos percatamos gracias a la salida anterior que sus curvaturas calculadas mediante la fórmula del círculo osculador a partir de la arista están muy próximos al valor 0,2. En realidad, el círculo rojo de la Figura 3.17 actúa como círculo osculador a partir de la arista E_4, ya que es tangente a las tres aristas sucesivas E_3, E_4 y E_5, por lo que parece lógico que su valor sea tan próximo al real, es decir, al de su círculo osculador. El hecho de que no sea exactamente 0,2 puede deberse al hecho de que el polígono no está representado y realizado con precisión.

En cuanto a las diferentes curvaturas obtenidas de cada uno de los otros vértices, advertimos como se aproximan (unos más que otros) al valor 0,2, pero sin ninguna reflexión importante que obtener de este hecho.

3.5. Conclusiones

Es posible elaborar la teoría de la **Geometría Diferencial Discreta** (DDG en inglés por las siglas *Discrete Differential Geometry*) de forma semejante a como se hace con la geometría diferencial diferenciable. Los análogos discretos de medidas geométricas como la longitud del arco, el tangente, la curvatura, etc. se pueden definir para que respeten al máximo (incluso de manera exacta) las propiedades de las variedades diferenciables.

No hay una opción única de cómo discretizar la curvatura. Como ejemplo claro, en los apartados anteriores hemos estudiado tres formas igualmente válidas de definir la curvatura discreta de curvas en el plano. DDG tiene como propósito el de prestar mucha atención a qué opciones conducen a qué discretizaciones, y cuáles son las consecuencias de estas elecciones.

Como consecuencia práctica, al elegir una discretización de curvatura para el cálculo, uno debe conocer las alternativas y sus ventajas y desventajas, y seleccionar una discretización de curvatura más adecuada para la tarea en particular.

Como aporte final, añado algunas conclusiones sacadas de [5]:

"*En cuanto a la curvatura obtenida a partir de los círculos osculadores a partir del vértice, la curvatura está limitada superiormente por 2, ya que el radio del círculo siempre será mayor o igual que* $\frac{1}{2}$*. Este problema estaría resuelto usando alguna de las otras dos definiciones de círculo osculador.*

En cuanto a la curvatura obtenida a partir de los círculos osculadores a partir de la arista, la curvatura no está limitada superiormente incluso para curvas parametrizadas por longitud de arco. En cambio, tiene la desventaja de no ser tan local como la definición anterior (está definido en las aristas las cuales dependen de las aristas vecinas, lo que lo hace involucrar un total de cuatro puntos consecutivos), mientras que solo es aplicable a curvas planas.

En cuanto a la curvatura obtenida a partir de los círculos osculadores a partir de curvas PLA, la cual utilizaba las curvas discretas parametrizadas por longitud de arco, podemos lograr la localidad de la primera y la no delimitación de la segunda definición haciendo que el círculo toque dos bordes consecutivos en sus puntos medios. Esto funciona en cualquier dimensión. En cambio, esta definición solamente es buena para curvas cuyos bordes tengan la misma longitud, ya que es una condición suficiente para la existencia de la tangente del círculo a ambos bordes en sus puntos medios."

Capítulo 4

Conclusiones generales

Este último capítulo se centrará en la descripción de las consideraciones más personales del proyecto de mi estancia en prácticas y de la memoria sobre Geometría Diferencial Discreta, puesto que dentro de cada capítulo ya se han desarrollado las conclusiones más formales y académicas de cada parte.

Por una parte, la estancia en prácticas ha sido una experiencia muy amena e instructiva en la que he aprendido como son tratadas las imágenes radiológicas de un centro médico. Tuve la oportunidad de establecerme una serie de objetivos que ir alcanzando poco a poco, pues estaba en un entorno bastante desconocido y necesitaba ir despacio. Me pude sentir realizado por ir alcanzando cada uno de esas metas hasta lograr llevar a cabo la realización de una aplicación utilizada como una herramienta de análisis en base a la información extraída de estudios e informes médicos radiológicos. Posteriormente, dicho programa será utilizada por la empresa en la que estuve realizando las prácticas para tomar decisiones empresariales y tener la posibilidad de extraer modelos de predicción.

Por otra parte, en cuanto al apartado matemático del trabajo final de grado, he de decir que me ha resultado muy interesante comprobar como se extrapolan a la realidad (entrono discreto) todos los conceptos dados en clase de geometría diferencial. Debido al amplio contenido de esta rama, solo he podido centrarme en una de las magnitudes más elementales de las curvas en el plano: la curvatura. He podido cerciorarme como todo lo dado en clase se complica en el entorno discreto. Me ha parecido fascinante el examinar la variedad de posibilidades de curvatura que se pueden determinar en la geometría diferencial discreta.

Bibliografía

[1] Actualmed Sistemas Radiológicos, *http://www.actualmed.com/*

[2] GPS Open Source, *https://www.gpsos.es/*

[3] J. Monterde, *Geometria Diferencial Clàssica*, Dpt. Geometria i Topologia Universitat de València, págs. 8-32, Curs 2008/2009.

[4] LUIS J. ALÍAS LINARES, *El significado geométrico de la curvatura*, Fundación Séneca, págs. 22-23, 2004.

[5] Alexander I. Bobenko, *Geometry II Discrete Differential Geometry*, págs. 5-15, 2015.

[6] Tim Hoffmann, *Discrete Differential Geometry of Curves and Surfaces*, Faculty of Mathematics, Kyushu University , págs. 8-18, 2008.

[7] A. V. Pogorélov, *Geometría elemental*, Editorial Mir Moscú , 1977.

[8] P. Puig Adam, *Curso de Geometría Métrica*, Tomo I. Fundamentos , 1986.

Anexo A

migracion.py

```
#!/usr/bin/python
import MySQLdb
import pydicom
from io import BytesIO
import pymongo
from pymongo import MongoClient
import datetime
from flask import Flask
import time
from flask import render_template


'''
################## EXTRACT DATA FROM MYSQL  ###################################
'''
# Query in SQL
query = "SELECT * \
          FROM files f JOIN instance  i ON f.instance_fk = i.pk \
                         JOIN series   sr ON i.series_fk   = sr.pk \
                         JOIN study    st ON sr.study_fk   = st.pk  \
                         JOIN patient  p ON st.patient_fk = p.pk;"

# Parameters of the database: host, user, password, database name
DB_HOST = 'localhost'
DB_USER = 'root'
DB_PASS = 'practicas123'
DB_NAME = 'pacs13'  # specify the database that you want to dump

# Create a database and a cursor objetct
db = MySQLdb.connect(DB_HOST, DB_USER, DB_PASS, DB_NAME)
dict_c = db.cursor(MySQLdb.cursors.DictCursor)


```

```
# Get data from database
try:
    dict_c.execute(query)
    data = dict_c.fetchall()
except MySQLdb.Error as e:
    try:
        print("MySQL Error [{0}]: {1}".format(e.args[0], e.args[1]))
    except IndexError as e:
            print("MySQL Error: {0}".format(str(e)))

# Close cursor and connection
dict_c.close()
db.close()


'''
################## EXTRACT DATA FROM DICOM  ###################################
'''
# List of data that we should not repeat with the attributes of DICOM
not_add = ["InstanceNumber", "SOPInstanceUID", "BodyPartExamined", "
    InstitutionName","Modality","PerformingPhysicianName", "SeriesDescription",
    "SeriesInstanceUID", "SeriesNumber", "StationName", "AccessionNumber", "
    ReferringPhysicianName", "StudyDescription", "StudyID", "StudyInstanceUID",
    "PatientID", "PatientName"]


# Function that extract data from sequence in DICOM
def sequence(item, lista):
    seq = item
    for ref in seq.dir():
        k = ref
        if "Sequence" in k:
            sequence(seq.get(k).pop(), lista)
        else:
            v = seq.get(k)
            if (k,v) not in lista:
                lista.append( (k, v) )

# Looping over a dictionary of data
for elem in data:
    # Lists where are stored the future additions and the future deletions
    post_add=[]
    post_del=[]
    # Looping over a elements of each of the data dictionaries
    for key, val in elem.items():
        # Special treatment beacuse the key in mongoDB must not contain '.'
        if "." in key:
            post_add.append( (key.replace(".", "_"), val) )
            post_del.append(key)
        # Special treatment beacuse mongoDB cannot encode object ''
        if val=='':
            post_add.append( (key, None) )
```

```
82          # Processing of DICOM files which contain the string "attrs"
83          if "attrs" in key:
84              post_del.append(key)
85              ds = pydicom.dcmread(BytesIO(val), force=True)
86              # Special treatment with DICOM that have different coding like '
                    ISO_2022_IR_6'
87              ds.SpecificCharacterSet = pydicom.charset.default_encoding
88              # Looping over a attributes of DICOM
89              for item in ds.dir():
90                  if item not in elem.keys() and item not in not_add:
91                      # Special treatment beacuse mongoDB cannot encode object ''
92                      if ds.get(item) == '':
93                          k = item
94                          v = None
95                      # Special treatment beacuse the key in mongoDB must not
                            contain '.'
96                      elif "." in item:
97                          k = item.replace(".", "_")
98                          v = ds.get(item)
99                          post_del.append(item)
100                     # Special treatment beacuse mongoDB cannot contain sequence
101                     elif "Sequence" in item:
102                         if len(ds.get(item)) > 0:
103                             sequence( ds.get(item).pop(0), post_add)
104                         continue
105                     # Special treatment beacuse mongoDB cannot encode this
                            array
106                     elif "Image" in item: #type(ds.get(item)) is list:
107                         k = item
108                         v = ds.get(item)[:]
109                     else:
110                         k = item
111                         v = ds.get(item)
112                     post_add.append( (k, v) )
113     # We do not need binary files anymore
114     for i in range(0, len(post_del)):
115         del(elem[post_del[i]])
116     # We need the data that are in DICOM but are not in the data dictionary
117     for i in range(0, len(post_add)):
118         k = post_add[i][0]
119         v = post_add[i][1]
120         elem[k] = v
121
122
123 client = MongoClient()
124 db = client.pacsdb
125 all_id = db.images.insert(data)
```

Anexo B

__init__.py

```
#!/usr/bin/python
import MySQLdb
import pydicom
from io import BytesIO
import pymongo
from pymongo import MongoClient
import datetime
from flask import Flask
import time
from flask import render_template
from bson.code import Code

reducer = Code("""
            function (obj, prev) {
        var ObjDT = new Date(obj.created_time);

        function redondear(numero, digitos){
            let base = Math.pow(10, digitos);
            let entero = Math.round(numero * base);
            total = entero / base;
            return total;
        }

        if (!isNaN(ObjDT.getTime())) {
            prev.fecha_ini = new Date(isNaN(prev.fecha_ini) ? obj.created_time
                : Math.min(prev.fecha_ini, obj.created_time));

            prev.fecha_fin = new Date(isNaN(prev.fecha_fin) ? obj.created_time
                : Math.max(prev.fecha_fin, obj.created_time));

            prev.dias = (prev.fecha_fin - prev.fecha_ini)/ (1000 * 3600 * 24);
            prev.dias = redondear(prev.dias, 2);

```

```
32              if(obj.study_iuid != null) prev.estudios += obj.study_iuid.length;
33              else prev.estudios++;
34
35              prev.estudiosDia =  (prev.dias != 0)? prev.estudios/prev.dias :
                    prev.estudios;
36              prev.estudiosDia = redondear(prev.estudiosDia, 2);
37
38
39              prev.mbytesTotales += obj.file_size;
40
41              prev.mbytesDia = (prev.dias != 0)? prev.mbytesTotales/prev.dias :
                    prev.mbytesTotales;
42              prev.mbytesDia = redondear(prev.mbytesDia, 2);
43
44
45              if(obj.series_iuid != null) prev.series += obj.series_iuid.length;
46              else prev.series++;
47
48
49              prev.instancias += obj.num_instances;
50
51
52              if(obj.ReferencedSOPInstanceUID != null) prev.instancias += obj.
                    ReferencedSOPInstanceUID.length;
53              else prev.instancias++;
54
55
56
57              prev.mbytesEstudios = prev.mbytesTotales/prev.estudios;
58              prev.mbytesEstudios = redondear(prev.mbytesEstudios, 2);
59
60              prev.mbytesSerie = prev.mbytesTotales/prev.series;
61              prev.mbytesSerie = redondear(prev.mbytesSerie, 2);
62
63              prev.mbytesInstancia = prev.mbytesTotales/prev.instancias;
64              prev.mbytesInstancia = redondear(prev.mbytesInstancia, 2);
65
66              prev.InstanciaSerie = prev.mbytesTotales/prev.series;
67              prev.InstanciaSerie = redondear(prev.InstanciaSerie, 2);
68          }
69             }
70              """)
71
72
73
74
75  '''
76 ################## LOAD DATA TO MONGODB  ###################################
77  '''
78  client = MongoClient()
79 db = client.pacsdb
80
```

```
81
82
83 '''
84 ################## FLASK  ###################################
85 '''
86 t_ini = time.time()
87 app = Flask(__name__)
88
89
90 i1 = db.images.create_index( "study_iuid")
91 i2 = db.images.ensure_index( "pat_id" )
92
93 # Creamos un decorator que nos indica que la ruta que va a ser definada es la
       raiz
94 @app.route("/index")
95 @app.route("/index/<todos>")
96 @app.route("/")
97 @app.route("/<todos>")
98 def index():
99     n_estudios = len(db.images.distinct("study_iuid"))
100     name_pac = DB_NAME
101     n_images = db.images.count()
102     n_patients = len(db.images.distinct("pat_id"))
103
104
105     modality = db.images.group(key={"modality":1}, condition={}, initial={"
          estudios":0, "mbytesTotales":0, "series":0, "instancias":0}, reduce=
          reducer)
106
107     n_modalities = len(db.images.distinct("modality"))
108
109
110
111
112
113
114     return render_template('index.html', todos= [n_estudios, name_pac, n_images
          , n_patients, modality]  )
```

Anexo C

index.html

```
<!DOCTYPE html>
<html>
    <head>
        <title>{% block title %}{% endblock %} Statistics </title>
        <style>
            table {
                font-family: arial, sans-serif;
                border-collapse: collapse;
                width: 100%;
            }

            td, th {
                border: 1px solid #dddddd;
                text-align: left;
                padding: 8px;
            }
        </style>
    </head>

    <body>

        <h2>ACTUALMED Table</h2>


    <table class="centered thick-border">
            <tr>
                    <th>Modalidad</th>
                    <th>Fecha inicio</th>
                    <th>Fecha fin</th>
                    <th>Dias</th>
                    <th>Estudios</th>
                    <th>Estudios/Dia</th>

```

```
34                      <th>Mbytes/dia</th>
35                      <th>Series</th>
36                      <th>Instancias</th>
37                      <th>Mbytes totales</th>
38                      <th>Mbytes/estudio</th>
39                      <th>Mbytes/serie</th>
40                      <th>Mbytes/instancia</th>
41                      <th>Instancia/serie</th>
42              </tr>
43              {% for mod in todos[4] %}
44                      <tr>
45                              <td>{{ mod['modality'] }}</td>
46                              <td>{{ mod['fecha_ini'] }}</td>
47                              <td>{{ mod['fecha_fin'] }}</td>
48                              <td>{{ mod['dias'] }}</td>
49                              <td>{{ mod['estudios'] }}</td>
50                              <td>{{ mod['estudiosDia'] }}</td>
51
52                              <td>{{ mod['mbytesDia'] }}</td>
53                              <td>{{ mod['series'] }}</td>
54                              <td>{{ mod['instancias'] }}</td>
55                              <td>{{ mod['mbytesTotales'] }}</td>
56                              <td>{{ mod['mbytesEstudios'] }}</td>
57                              <td>{{ mod['mbytesSerie'] }}</td>
58                              <td>{{ mod['mbytesInstancia'] }}</td>
59                              <td>{{ mod['InstanciaSerie'] }}</td>
60
61
62                      </tr>
63              {% endfor %}
64      </table>
65
66      </body>
67  </html>
```

Anexo D

ComparacionCurvaturas.m

```
1
2 function ComparacionCurvaturas
3
4
5    % Defino una circunferencia en coordenadas polares
6     radio = 2;
7     t_cir = 0:pi/50:2*pi;
8     x_cir = radio*sin(t_cir);    y_cir = radio*cos(t_cir);
9
10
11   % Defino un poligono regular inscrito en la circunferencia
12    lados = 6;
13    t_pol = 0:2*pi/lados:2*pi;
14    x_pol = radio*sin(t_pol);    y_pol = radio*cos(t_pol);
15
16
17   % Determino los vertices del poligono
18    for i = 1:length(x_pol)
19        V_pol(i,1) = x_pol(i);
20        V_pol(i,2) = y_pol(i);
21    end
22
23
24   % Como el poligono es regular, cada angulo mide
25    ang_grad = ((lados-2)*180)/lados;
26    ang = pi - (ang_grad*pi)/180;
27
28
29   % Calculo la curvatura de la circunferencia
30    k_cir = 1/radio;
31
32
33
```

```
34    %Calculo las curvaturas en cada vertice del poligono con las formulas
35     for i = 1:length(V_pol)-1
36        %Establecimiento de vertices
37         if i==1
38            v_ant = V_pol(length(V_pol)-1,:);
39            v_act = V_pol(i,:);
40            v_pos = V_pol(i+1,:);
41         else
42            v_ant = V_pol(i-1,:);
43            v_act = V_pol(i,:);
44            v_pos = V_pol(i+1,:);
45         end
46
47        %a partir del vertice
48         ki_ver(i) = (2*sin(ang)) / norm(v_pos - v_ant );
49
50        %a partir de la arista
51         ki_ari(i) = (tan(ang/2)+tan(ang/2)) / norm(v_pos - v_act );
52
53        %a partir de curvas PLA
54         ki_pla(i) = 2*tan(ang/2);
55
56     end
57
58
59    %SALIDA por pantalla
60     disp(['Curvatura del circulo: ', num2str( k_cir )])
61     disp(' ')
62     disp(['Curvatura a partir de los vertices: ', num2str( ki_ver(1) )])
63     disp(['Curvatura a partir de las aristas:   ', num2str( ki_ari(1) )])
64     disp(['Curvatura a partir de curvas PLA:    ', num2str( ki_pla(1) )])
65
66
67    %Graficas
68     hold on;
69     plot(x_cir, y_cir, 'r') %Circunferencia
70     plot(x_pol, y_pol, 'b') %Poligono
71     axis equal;
72
73end
```

Anexo E

ComparacionCurvaturasIrregular.m

```
1
2 function ComparacionCurvaturasIrregular
3
4    % Defino una circunferencia en coordenadas polares
5     radio = 5;
6     t_cir = 0:pi/50:2*pi;
7     x_cir = radio*sin(t_cir);    y_cir = radio*cos(t_cir);
8
9    % Defino un poligono irregular
10    x_pol = [0   4   5    5   2.14   -3   -6   -2   -5   -3   0];
11    y_pol = [5   3   2   -2    -5    -5    -3   -1   2   4   5];
12
13   % Determino los vertices del poligono
14    for i = 1:length(x_pol)
15        V_pol(i,1) = x_pol(i);
16        V_pol(i,2) = y_pol(i);
17    end
18
19   % Determino los vectores de las aristas (orientacion agujas del reloj)
20    for i = 1:length(V_pol)-1
21        % Establecimiento de vertices
22        if i==1
23           v_act = V_pol(i,:);
24           v_pos = V_pol(i+1,:);
25        else
26           v_act = V_pol(i,:);
27           v_pos = V_pol(i+1,:);
28        end
29
30        vector(i,1) = v_pos(1) - v_act(1);
31        vector(i,2) = v_pos(2) - v_act(2);
32    end
33
```

```
34
35
36    %Determino los angulos entre cada par de aristas del poligono
37     for i = 1:length(V_pol)-1
38         %Establecimiento de vertices
39         if i==1
40            u = vector(length(V_pol)-1,:);
41            v = vector(i,:);
42         else
43            u = vector(i-1,:);
44            v = vector(i,:);
45         end
46
47         cos_alpha(i) = abs( u(1)*v(1) + u(2)*v(2) ) / ( sqrt(u(1)^2+u(2)^2) *
               sqrt(v(1)^2+v(2)^2));
48         ang(i) = acos(cos_alpha(i)); %radianes
49         %ang_grad(i) = (ang(i)*180)/pi;
50
51     end
52
53
54    %Calculo la curvatura de la circunferencia
55     k_cir = 1/radio;
56
57
58    %Calculo las curvaturas en cada vertice del poligono con las formulas
59     for i = 1:length(V_pol)-1
60        %Establecimiento de vertices
61         if i==1
62            v_ant = V_pol(length(V_pol)-1,:);
63            v_act = V_pol(i,:);
64            v_pos = V_pol(i+1,:);
65         else
66            v_ant = V_pol(i-1,:);
67            v_act = V_pol(i,:);
68            v_pos = V_pol(i+1,:);
69         end
70
71        %a partir del vertice
72         ki_ver(i) = (2*sin(ang(i))) / norm(v_pos - v_ant );
73
74        %a partir de la arista
75         ki_ari(i) = (tan(ang(i)/2)+tan(ang(i)/2)) / norm(v_pos - v_act );
76
77        %a partir de curvas PLA
78         ki_pla(i) = 2*tan(ang(i)/2);
79
80     end
81
82
83
84
```

```
85    %SALIDA por pantalla
86     disp(['Curvatura del circulo: ', num2str( k_cir )])
87     disp(' ')
88     for i = 1:length(V_pol)-1
89        disp(['Vertice ',num2str(i),':'])
90        disp(['    Curvatura a partir de los vertices: ', num2str( ki_ver(i) )])
91        disp(['    Curvatura a partir de las aristas: ', num2str( ki_ari(i) )])
92        disp(['    Curvatura a partir de curvas PLA: ', num2str( ki_pla(i) )])
93        disp(' ')
94     end
95
96
97    %Graficas
98     hold on;
99     plot(x_cir, y_cir, 'r') %Circunferencia
100    plot(x_pol, y_pol, 'bo-') %Poligono
101    axis equal;
102
103end
```