Apéndice B: Código de Algoritmo de Procesamiento de Imágenes

Apéndice B.I: Código de Procesamiento de Formas

```
-(NSMutableArray *)analizarForma
{
    int counter = 0;
    int similarPixels = 0;
    int maximo = 0;
    int posicion = 0;
    int numeroImagenes = 0;

    UIImage *imagenAnalizar = [self cargarImagen];

    NSMutableArray *imagenes = [self recuperarImagenesCanny];
    NSMutableArray *imagenesComparadas = [[NSMutableArray alloc] init];
    NSMutableArray *similitudes = [[NSMutableArray alloc] init];
    NSMutableArray *resultado = [[NSMutableArray alloc] init];
    NSMutableArray *similitudesResultado = [[NSMutableArray alloc] init];
    identificadoresForma = [[NSMutableArray alloc] init];

    numeroImagenes = [imagenes count];

    cv::Mat matAnalizar = [self MatFromUIImageColor:imagenAnalizar];
    cv::Mat cannyComparar;

    cv::Mat grayAnalizar;
    cv::cvtColor(matAnalizar, grayAnalizar, CV_RGB2GRAY);

    cv::Mat cannyAnalizar;
    cv::Mat result;
    cv::Canny(grayAnalizar, cannyAnalizar, 150, 250);

    while (counter < numeroImagenes){
        cannyComparar = [self MatFromUIImageGray:imagenes[counter]];
        cv::compare(cannyAnalizar, cannyComparar, result, cv::CMP_EQ);
        similarPixels = cv::countNonZero(result);
        NSNumber *similitud = [NSNumber numberWithInt:similarPixels];
        UIImage *imagenResultado = [self UIImageFromCVMat:cannyComparar];
        [imagenesComparadas addObject:imagenResultado];
        [similitudes addObject:similitud];
        counter++;
```

```
    }
    if ([imagenesComparadas count] == 0)
    {
        return nil;
    }
    else{
        for (int i = 0; i < 4; i++){
            maximo = [[similitudes valueForKeyPath:@"@max.self"] intValue];
            NSNumber *max = [NSNumber numberWithInt:maximo];
            [similitudesResultado addObject:max];
            posicion = [similitudes indexOfObject:max];
            UIImage *imagenResultado = imagenesComparadas[posicion];
            [resultado addObject:imagenResultado];
            [similitudes removeObjectAtIndex:posicion];
            int idDB = posicion + 1;
            NSNumber *imageID = [NSNumber numberWithInt:idDB];
            [identificadoresForma addObject:imageID];
        }
    }
    NSLog(@"%@", similitudesResultado);
    NSLog(@"%@", identificadoresForma);
    return resultado;
}
```

Apendice B.II: Código de Procesamiento de Colores

```
-(NSMutableArray *)analizarColor
{
    int counter = 0;
    int h_bins = 50;
    int s_bins = 60;
    int histSize[] = {h_bins, s_bins};
    int channels[] = {0, 1};
    int posicion = 0;
    double maximo = 0;
    int numeroImagenes = 0;

    float h_range[] = {0, 256};
    float s_range[] = {0, 180};
    const float *range[] = {h_range, s_range};

    UIImage *imagenAnalizar = [self cargarImagen];

    NSMutableArray *imagenes = [self recuperarImagenesColor];
    NSMutableArray *imagenesComparadas = [[NSMutableArray alloc] init];
    NSMutableArray *similitudes = [[NSMutableArray alloc] init];
    NSMutableArray *resultado = [[NSMutableArray alloc] init];
    NSMutableArray *similitudesResultado = [[NSMutableArray alloc] init];
    identificadoresColor = [[NSMutableArray alloc] init];
```

```
    numeroImagenes = [imagenes count];

    cv::Mat matAnalizar = [self MatFromUIImageColor:imagenAnalizar];
    cv::Mat matComparar;
    cv::Mat matHSVAnalizar;
    cv::Mat matHSVComparar;

    cv::MatND hist_Analizar;
    cv::MatND hist_Comparar;

    cv::cvtColor(matAnalizar, matHSVAnalizar, CV_RGB2HSV);
    cv::calcHist(&matHSVAnalizar, 1, channels, cv::Mat(), hist_Analizar, 2, histSize, range,
true, false);
    cv::normalize(hist_Analizar, hist_Analizar, 0, 1, cv::NORM_MINMAX, -1, cv::Mat());
    while (counter < numeroImagenes){
        matComparar = [self MatFromUIImageColor:imagenes[counter]];
        cv::cvtColor(matComparar, matHSVComparar, CV_RGB2HSV);
        cv::calcHist(&matHSVComparar, 1, channels, cv::Mat(), hist_Comparar, 2, histSize,
range, true, false);
        cv::normalize(hist_Comparar, hist_Comparar, 0, 1, cv::NORM_MINMAX, -1,
cv::Mat());
        double sim = cv::compareHist(hist_Analizar, hist_Comparar, CV_COMP_CORREL);
        NSNumber *similitud = [NSNumber numberWithDouble:sim];
        cv::cvtColor(matHSVComparar, matComparar, CV_HSV2RGB);
        UIImage *imagenResultado = [self UIImageFromCVMat:matComparar];
        [imagenesComparadas addObject:imagenResultado];
        [similitudes addObject:similitud];
        counter++;
    }
    if ([imagenesComparadas count] == 0){
        return nil;
    }
    else{
        for (int i = 0; i < 4; i++){
            maximo = [[similitudes valueForKeyPath:@"@max.self"] doubleValue];
            NSNumber *max = [NSNumber numberWithDouble:maximo];
            [similitudesResultado addObject:max];
            posicion = [similitudes indexOfObject:max];
            UIImage *imagenResultado = imagenesComparadas[posicion];
            [resultado addObject:imagenResultado];
            int idDB = posicion + 1;
            NSNumber *imageID = [NSNumber numberWithInt:idDB];
            similitudes[posicion] = [NSNumber numberWithInt:-1];
            [identificadoresColor addObject:imageID];
        }
    }
    NSLog(@"%@", similitudesResultado);
    NSLog(@"%@", identificadoresColor);
    return resultado;

}
```

Apendice B.III: Código de Procesamiento de Resultados de los Primeros Dos Pasos

```
-(void)analizarFormaColor
{
    NSMutableArray *resultadoForma = [self analizarForma];
    NSMutableArray *resultadoColor = [self analizarColor];
    NSMutableArray *imagenesComparadas = [[NSMutableArray alloc] init];
    NSMutableArray *imagenesResultado = [[NSMutableArray alloc] init];
    NSMutableArray *similitudes = [[NSMutableArray alloc] init];
    NSMutableArray *similitudesResultado = [[NSMutableArray alloc] init];
    identificadoresFinal = [[NSMutableArray alloc] init];
    identificadoresResultado = [[NSMutableArray alloc] init];

    int numeroForma = 0;
    int numeroColor = 0;

    if (([resultadoForma count] == 0) && ([resultadoColor count] == 0)) {
        UIAlertView *alert = [[UIAlertView alloc ] initWithTitle:@"Error" message:@"Lo
Sentimos, No Se Encontro Información Sobre Dicha Imagen" delegate:self
cancelButtonTitle:@"Cerrar" otherButtonTitles:nil, nil];
        [alert show];
    }
    else{
        cv::Mat forma;
        cv::Mat color;
        cv::Mat resultado;

        int similitud = 0;


        cv::Mat grayColor;
        cv::Mat cannyColor;

        for (int y = 0; y < 4; y++){
            for (int x = 0; x < 4; x++){
                if (identificadoresForma[y] == identificadoresColor[x]){
                    [identificadoresResultado addObject:identificadoresColor[x]];
                }
            }

        }
        NSLog(@"%@", identificadoresResultado);
```

```
for (int z = 0; z < [identificadoresResultado count]; z++){
    int indexColor;
    int indexForma;
    indexColor = [identificadoresColor indexOfObject:identificadoresResultado[z]];
    indexForma = [identificadoresForma indexOfObject:identificadoresResultado[z]];
    [imagenesResultado addObject:resultadoColor[indexColor]];
    [identificadoresColor removeObjectAtIndex:indexColor];
    [identificadoresForma removeObjectAtIndex:indexForma];
    [resultadoColor removeObjectAtIndex:indexColor];
    [resultadoForma removeObjectAtIndex:indexForma];
}
NSLog(@"%@", identificadoresColor);
NSLog(@"%@", identificadoresForma);

numeroForma = [resultadoForma count];
numeroColor = [resultadoColor count];

for (int i = 0; i < numeroForma; i++){
    forma = [self MatFromUIImageGray:resultadoForma[i]];
    for (int j = 0; j < numeroColor; j++){
        color = [self MatFromUIImageColor:resultadoColor[j]];
        cv::cvtColor(color, grayColor, CV_RGB2GRAY);
        cv::Canny(grayColor, cannyColor, 150, 250);
        cv::compare(forma, cannyColor, resultado, cv::CMP_EQ);
        similitud = cv::countNonZero(resultado);
        NSNumber *sim = [NSNumber numberWithInt:similitud];
        [similitudes addObject:sim];
        [imagenesComparadas addObject:resultadoColor[j]];
        NSNumber *imageID = identificadoresColor[j];
        [identificadoresFinal addObject:imageID];
    }
}
NSLog(@"%@", identificadoresFinal);
NSLog(@"%@", similitudes);

int maximo = 0;
int posicion = 0;
int numeroComparar = [imagenesComparadas count];
NSNumber *max;
for (int k = 0; k < numeroComparar; k++){
    maximo = [[similitudes valueForKeyPath:@"@max.self"] intValue];
    max = [NSNumber numberWithInt:maximo];
    [similitudesResultado addObject:max];
    posicion = [similitudes indexOfObject:max];
    [imagenesResultado addObject:imagenesComparadas[posicion]];
    similitudes[posicion] = [NSNumber numberWithInt:-1];
    [identificadoresResultado addObject:identificadoresFinal[posicion]];
}
NSLog(@"%@", similitudesResultado);
NSLog(@"%@", imagenesResultado);
NSLog(@"%@", identificadoresResultado);
```

```
    }
    NSString *result = [NSString stringWithFormat:@"%@", identificadoresResultado[0]];
    NSString *result2 = [result stringByAppendingString:@"_"];
    NSString *result3 = [result2 stringByAppendingString:[NSString
stringWithFormat:@"%@", identificadoresResultado[1]]];
    NSString *result4 = [result3 stringByAppendingString:@"_"];
    NSString *resultFinal = [result4 stringByAppendingString:[NSString
stringWithFormat:@"%@", identificadoresResultado[2]]];
    [self saveResult:resultFinal];
}

-(void)saveResult:(NSString *)resultado
{
    [UIApplication sharedApplication].networkActivityIndicatorVisible = YES;
    NSString *consulta = [[NSString alloc] initWithFormat:@"Resultado=%@", resultado];
    NSData *dataConsulta = [consulta dataUsingEncoding:NSUTF8StringEncoding];
    NSString *urlString = @"http://192.168.1.106/tesis/saveResults.php";

    NSMutableURLRequest *request = [[NSMutableURLRequest alloc] init];
    [request setURL:[NSURL URLWithString:urlString]];
    [request setHTTPMethod:@"POST"];

    NSMutableData *body = [NSMutableData data];
    [body appendData:dataConsulta];

    [request setHTTPBody:body];

    NSData *returnData = [NSURLConnection sendSynchronousRequest:request
returningResponse:nil error:nil];
    NSString *result = [[NSString alloc] initWithData:returnData
encoding:NSUTF8StringEncoding];
    NSLog(@"%@", result);
    [UIApplication sharedApplication].networkActivityIndicatorVisible = NO;
}
```

X

Dr. Oleg Starostenko Basarab
Asesor de Tesis