

## ANEXO I

### Metodología para la construcción de un cuadrante

Pese a ser la unidad muestral más ampliamente utilizada en los muestreos directos de *P. Oceanica* (ver referencia [30] en la Memoria), existen escasas referencias sobre dónde conseguir un cuadrante de estas características o cómo construirlo. Con el objetivo de facilitar la tarea a futuros muestreadores, a continuación se detalla un procedimiento para construir un cuadrante con materiales bajo coste y en 5 sencillos pasos.

En primer lugar, deberemos reunir los siguientes materiales y herramientas (Figura I):

- 1 tubo de PVC de color blanco, de 2 cm de diámetro y 2 m de longitud.
- 4 codos de PVC de 2 cm de diámetro.
- Sierra.
- Pistola de calar.
- Broca.
- Cinta métrica y lápiz.
- Lija.
- Cola apta para PVC o cinta aislante de color amarillo (opcional).



*Figura I. Materiales necesarios para la construcción de un cuadrante.*

En algunas grandes superficies de suministros de ferretería se puede acceder a una mesa de trabajo con las herramientas anteriormente indicadas, lo que facilita la tarea.

Pasos a seguir:

1. **Marcar** en el tubo 4 segmentos de 40 cm de largo.
2. **Cortar** con ayuda de la sierra. Es importante fijar el tubo con unas abrazaderas a la mesa para garantizar nuestra seguridad.
3. **Lijar** los extremos de los segmentos de tubo.
4. **Agujerear** cada segmento al menos un par de veces. Esto permitirá que el agua entre una vez montado y el cuadrante se hunda, de lo contrario flotaría.
5. **Montar** el cuadrante. En función de las necesidades del muestreo, se puede optar por fijar los tubos a los codos con cinta aislante (cuadrante desmontable) o fijar con cola apta para PVC (cuadrante fijo).

## ANEXO II

### Script de Matlab® que convierte los datos de formato .txt a .mat.

```
%Este archivo lee .txt Exportados de Sonar5_pro

clear all

%Abre el cuadro de dialogo para que se elija el .txt

[filename,pathname]=uigetfile('*.txt','Select txt data file');
fname=[pathname filename];

%Lee de la cabecera el número de pings que contiene el archivo
Rping = textread(fname,'%s',6, 'headerlines', 19);
pingini=str2num(Rping{3});
pingfin=str2num(Rping{6});
npings=pingfin-pingini;

%Lee de la cabecera el número de columnas donde se almacena el rango que contiene el
archivo
Rrange = textread(fname,'%s',11, 'headerlines', 24);
aux=char(Rrange(3));
aux=aux(1:5);
rangeini=str2num(aux);
aux2=char(Rrange(6));
aux2=aux2(1:5);
rangefin=str2num(aux2);
range=rangefin-rangeini;
mps=str2num(Rrange{11});
%asi es como deberia calcularse a la espera de que conteste Helge
%nrange=range/mps;
%nrange=nrange+7;

%Lee los datos
Rdata = textread(fname,'%s', 'headerlines', 32);

%Calcula el número de columnas
nrange=length(Rdata)/npings;

%Construye la matriz de tipo cell
TMS= zeros(npings,1,'double');
i=1;
fin=round(nrange);
fin2=fin;
for j=1:npings

    auxR=Rdata(i:fin2);
    R(j,:)=auxR;
    T= auxR{2};
    H= str2double(T(1:2));
    M= str2double(T(4:5));
```

```
S= str2double(T(7:8));
MS= str2double(T(10:size(T,2)));
TMS(j,1)= (H*3600+M*60+S)*1000+MS; %Guarda tiempo en ms
i=i+fin;
fin2=fin2+fin;
end

%Elimina de la matriz los datos que no son de interés
nrange=round(nrange);
RR=R(:,5:nrange);

%Convierte la matriz final de cell a formato numerico
for k=1:nrange-5
    for l=1:npings
        RR2(l,k)=str2num(RR{l,k});
    end
end

%Construye vector con el número de los pings
ax=pingini;
for k=1:npings
    x(k)=ax;
    ax=ax+1;
end

%Construye el vector con el número de divisiones espaciales, rango
auxy=range/nrange;
y=(rangeini:auxy:rangefin-auxy);

%Eliminamos el minimo y lo sustituimos por el minimo valor detectado
[fil col]=size(RR2);
for i=1:fil
    for j=1:col
        if RR2(i,j)==-900
            RR2(i,j)=-80;
        end
    end
end

ais=min(min(RR2));
for i=1:fil
    for j=1:col
        if RR2(i,j)==-80
            RR2(i,j)=ais;
        end
    end
end

%guarda como .mat
%save(filename);
save([filename(1:size(filename,2)-4) '.mat'], 'pingini', 'npings', 'x', 'y', 'RR2', 'TMS');

%Representa el ecograma
figure, imagesc(x,y,RR2), colorbar,
xlabel('Nº ping'), ylabel('R (m)')
```

## ANEXO III

### Script de Matlab® del algoritmo de detección de fondo

```

close all, clear all, clc
% Lectura del fichero
load('posidonia_0_graus_2016.mat')
% HS: configuración en cada ping
% P: ecograma
% PS: hora y coordenadas
sv=RR2';
[m, n]=size(sv);
% Aplicación de función TVG -----
for pingno=1:n
    %s'han d'extraure estes dades del sonarPro directament. En el fitxer
    %instruccions2.docx tens d'on es trauen. Les que he posat aquí són les
    %teues.
    frequency=200000;
    transmitpower=100;
    pulselength=0.000064;
    sampleinterval=pulselength/975;
    soundvelocity=1537;
    absorptioncoefficient=89.97;
    %samplespace = soundvelocity.*sampleinterval/2;
    samplespace = 0.0123;
    lambda = soundvelocity/frequency;
    % Sv values amb diferents tvg. Nomes si volem treballar amb un tvg
diferent. Pensar per sonar pro amb TVG DE 20log
    svcenterrange = soundvelocity.*pulselength/4;
    svrange = (0:m-1)'*samplespace - svcenterrange;
    svtvgrange = svrange;
    svtvgrange(find(svtvgrange<samplespace)) = samplespace;
    tvg40 = 40*log10(svtvgrange) + 2*absorptioncoefficient*svtvgrange;
    idxtvg40less0 = find(tvg40<0);
    tvg40(idxtvg40less0) = 0;

%svconst=10*log10(transmitpower*lambda^2*soundvelocity*pulselength/(32*pi^2
))...+ 2*(gain+sacorreccion) + psi;
    P(:,pingno) = sv(:,pingno) +tvg40; %- svconst; % P és la variable
d'scattering SENSE corregir en profunditat. Li podríem aplicar un nou tvg
des de matlab
    svcenterrange = soundvelocity.*pulselength/4;
    svrange = (0:m-1)'*samplespace - svcenterrange;
    svtvgrange = svrange;
    svtvgrange(find(svtvgrange<samplespace)) = samplespace;
    tvg20 = 20*log10(svtvgrange) + 2*absorptioncoefficient*svtvgrange;
    idxtvg20less0 = find(tvg20<0);
    tvg20(idxtvg20less0) = 0;
    sv20(:,pingno) = P(:,pingno) -tvg20;

    %si vull TVG30
    svcenterrange = soundvelocity.*pulselength/4;
    svrange = (0:m-1)'*samplespace - svcenterrange;
    svtvgrange = svrange;
    svtvgrange(find(svtvgrange<samplespace)) = samplespace;
    tvg30 = 30*log10(svtvgrange) + 2*absorptioncoefficient*svtvgrange;
    idxtvg30less0 = find(tvg30<0);
    tvg30(idxtvg30less0) = 0;
    sv30(:,pingno) = P(:,pingno) -tvg30;
end
% figure, imagesc(1:n,(0:m-1)*samplespace,P, [-160 20]), colorbar,

```

```

% xlabel('Ping number'), ylabel('Depth (m)')
%sv=sv20;
figure, imagesc(1:n,(0:m-1)*samplespace,sv), colorbar,
xlabel('Nº ping'), ylabel('R (m)')
% Detección del fondo -----
% Para cada ping se localiza el máximo y se le resta la duración del
% pulso emitido
pingno
r0=round(1/samplespace); % distancia inicial, en nº muestras
power=zeros(pingno,1);
ix=zeros(pingno,1);
for pingno=1:n
    [power(pingno,1), ix(pingno,1)]=max(sv(r0:end, pingno));
end
% ix=ix+r0-1;
% ix_b=ix-round(pulselength/sampleinterval);
% fondo=ix_b*samplespace;
% figure, plot(1:n,fondo), xlabel('Ping number'), ylabel('Depth (m)')

fondo=1+ix*samplespace;
fondo_av=zeros(length(fondo)-9,1);
for i=1:length(fondo)-9
    fondo_av(i)=sum(fondo(i:i+9))/10;
end
figure, plot(1:n,fondo), xlabel('Ping number'), ylabel('Depth (m)')
figure, plot(5:(length(fondo_av)+4),fondo_av), xlabel('Ping number'),
ylabel('Depth ave(m)')

%Calculo de la altura de posidonia
hpos=zeros(n,1);hpos_av=zeros(n-9,1);
inpos=zeros(n,1);
for pingno=1:n
    inpos(pingno)=ix(pingno);

    limit_h=find(sv((r0:end),pingno)>=(power(pingno,1)-30));
    inpos(pingno)=min(limit_h);
    %[power(pingno,1), ix(pingno,1)]=max(sv(r0:end, pingno));
end
inpos_av=zeros(length(inpos)-9,1);
for i=1:length(inpos)-9
    inpos_av(i)=sum(inpos(i:i+9))/10;
end
hpos=fondo-(1+inpos*samplespace);
hpos_av=fondo_av-(1+inpos_av*samplespace);
figure,plot(1:n,hpos), xlabel('Ping number'), ylabel('Alçada posidonia
(m)')
figure,plot(5:n-5,hpos_av), xlabel('Ping number'), ylabel('Alçada posidonia
average (m)')
%traure promig entre pingA i ping B
pingA=1;
pingB=n;
h_ave=sum(hpos(pingA:pingB))/(pingB-pingA)
% % Representación pings -----

figure, plot((0:m-1)*samplespace,sv(:,98), (0:m-1)*samplespace,sv(:,98))

xlabel('Depth (m)'), ylabel('Echo level (dB)')
legend('initial','end')
%% Integración de la energía -----
Pl=10.^(sv/10);

```

```

E=P1*sampleinterval;
% Curva de energía acumulada en toda la columna de agua
Eacum=zeros(m-r0+1,n);
for pingno=1:n
    Eacum(:,pingno)=cumsum(E(r0:end,pingno).^2);
end
%figure, hold all
% for pingno=1:10:n
% plot((r0:m)*samplespace,Eacum(:,pingno)/max(Eacum(:,pingno)))
% end
%xlabel('Depth (m)'), ylabel('Cumulative energy normalized')
%Alçada posidònia via Energia acumulada
hposE=zeros(n,1);
for pingno=1:n
    inpos=ix(pingno);
    % for in=r0:m
    %     if (sv(in,pingno)<0.1*max(Eacum(:,pingno)))
    %         inpos=inpos;
    %     elseif (sv(in,pingno)>=0.1*max(Eacum(:,pingno)))
    %         if (in<inpos)
    %             inpos=in;
    %         else
    %             inpos=inpos;
    %         end
    %     end
    % end
limit_E1=find(Eacum(:,pingno))>=0.1*max(Eacum(:,pingno));
inpos=min(limit_E1);
fons_E1=find(Eacum(:,pingno))>0.95*max(Eacum(:,pingno));
fons_E=min(fons_E1);
hposE(pingno,1)=(fons_E-inpos)*samplespace;

%[power(pingno,1), ix(pingno,1)]=max(sv(r0:end, pingno));
end
figure,plot(1:n,hpos), xlabel('Ping number'), ylabel('Alçada posidonia per
energia (m)')
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Energía acumulada en el fondo
m_fondo=round(1/1500/samplespace);
E_fondo=zeros(1,n);
for pingno=1:n
    % E_fondo(pingno)=sum(E(ix_b(pingno)-m_fondo:ix_b(pingno),pingno));
    E_fondo(pingno)=sum(E(ix(pingno)-m_fondo:ix(pingno),pingno));
end;

% forma_fondo=hilbert(E_fondo'); %forma de onda con la transformada
deHilbert
% figure, plot(abs(forma_fondo),[1:pingno])
% xlabel('Bottom energy'),ylabel('ping')

% Energía acumulada en el metro y medio anterior al fondo
r_limit=round(1.5/samplespace);
Eacumpos=zeros(n,1);
Elogpos=zeros(n,1);
for pingno=1:n
    Eacumpos(pingno,1)=sum(E((ix(pingno)-r_limit):ix(pingno),pingno).^2);
    Elogpos=10*log10(Eacumpos);
end
figure, plot([1:pingno],Eacumpos)
xlabel('Energia acumulada posidònia'),ylabel('ping')

```

```
figure, plot([1:pingno],Elogpos)  
xlabel('Energia acumulada posidònia (dB)'),ylabel('ping')
```