

# Algorithms for Fast Convolutions on Motion Groups

Alexander B. Kyatkin and Gregory S. Chirikjian

*Department of Mechanical Engineering, Johns Hopkins University, Baltimore, Maryland 21218*

*Communicated by Stephane Jaffard*

Received December 22, 1998; revised December 15, 1999

---

In this paper we apply techniques from noncommutative harmonic analysis to the development of fast algorithms for the computation of convolution integrals on motion groups. In particular, we focus on the group of rigid-body motions in 3-space, which is denoted here as SE(3). The general theory of irreducible unitary representations (IURs) of the 3D motion group is described briefly. Using IURs in operator form, we write the Fourier transform of functions on the motion group as an integral over the product space  $\text{SE}(3) \times S^2$ . The integral form of the Fourier transform matrix elements allows us to apply fast Fourier transform (FFT) methods developed previously for  $\mathbb{R}^3$ ,  $S^2$ , and  $\text{SO}(3)$  to speed up considerably the computation of convolutions of functions on SE(3). Such convolutions have been shown to play an important role in a number of engineering disciplines. An algorithm for the fast computation of the Fourier transform is given and its complexity is discussed. The Fourier transform for the 3D “discrete motion group” (semi-direct product of the icosahedral group with the translation group) is also developed and the computational complexity is discussed. © 2000 Academic Press

---

## 1. INTRODUCTION

In this paper we apply noncommutative harmonic analysis to the development of fast numerical algorithms for the computation of convolution integrals on the motion group of three-dimensional Euclidean space. This group, which is often called the special Euclidean group, will be denoted throughout this paper as SE(3). Convolution integrals on SE(3) arise in a number of scenarios.

Our motivation for deriving a fast Fourier transform (FFT) for the motion group came from the need to compute convolutions on the motion group efficiently. In a series of papers, the importance of motion-group convolutions in robotics [4], polymer science [5], and image analysis [17, 19] has been established. Our initial interest in convolution-like operations arose in the concrete context of manipulator workspace generation [14]. It was only later that we realized that the numerical procedures we were implementing were essentially motion-group convolutions.

Other applications and connections between noncommutative harmonic analysis and classical Fourier analysis can be found in [6]. The common feature in all of the applications

of motion-group convolutions studied in engineering to date is that one function is rigidly swept (rotated and translated) over another. From a practical perspective, motion-group convolutions are very much like convolutions on the line or circle, since the motion groups are “almost like”  $\mathbb{R}^n$ . The most significant difference is that whereas translations commute, rigid-body motions do not. In addition to the practical problems in which such convolutions arise, the motion groups serve as the first example of a class of noncompact noncommutative Lie groups for which FFTs can be defined when dealing with compactly supported functions. We note also that the motion groups are not nilpotent or semisimple, which are, in a sense, the easiest kinds of noncompact noncommutative Lie groups to study.

Below, the general theory of irreducible unitary representations (IURs) of the motion group are described briefly. We show how to write the Fourier transform on  $SE(3)$  using matrix elements of IURs. The Fourier transform is written in an integral form which allows us to apply (after interpolation from a rectangular grid to a spherical coordinate grid), fast Fourier transform methods for  $\mathbb{R}^3$  and for the 2-sphere. For  $N = O(S^6)$  sample points ( $S$  being a representative number of samples in each coordinate parameterizing the group), the complexity of computations is reduced from  $O(N^2)$  to  $O(N^{7/6}(\log N)^2) + O(N^{(\gamma+1)/3})$  where the second term depends on the complexity of matrix multiplication, and  $2 \leq \gamma \leq 3$  depending on how matrix multiplication is implemented.

We also develop Fourier transform methods for the “discrete motion group”, which is the semi-direct product of the continuous translation group and any finite subgroup of  $SO(3)$  (the icosahedral group in our example). We discuss briefly the computational complexity of the numerical implementation of the Fourier transform for the discrete motion group.

In Section 2 we give the general theory behind the Fourier transform of functions on  $SE(3)$ . An algorithm for fast computation of the Fourier transform of functions on  $SE(2)$  and  $SE(3)$  is given in Section 3. Section 4 describes the Fourier transform on the discrete motion group and its computational complexity.

## 2. FOURIER TRANSFORM FOR THE 3D MOTION GROUP

We describe in this section the Fourier transform of functions on the motion group which allows one to write convolution integrals on the motion group as a matrix product in Fourier space. Convolution on the motion group is defined as

$$(f_1 * f_2)(g) = \int_G f_1(h) f_2(h^{-1} \circ g) d\mu(h), \quad (1)$$

where the integral is taken over the group with respect to the measure  $\mu$ . We remind the reader that the motion group  $G = SE(3)$  is *unimodular*; i.e., it has a measure  $\mu$  such that if

$$\mu(f) = \int_G f(g) d\mu(g),$$

then  $\mu(R(h)f) = \mu(L(h)f) = \mu(f)$  where  $R(h)f(g) = f(g \circ h)$  and  $L(h)f(g) = f(h^{-1} \circ g)$  are respectively right- and left-shifted versions of the function  $f(g)$  with

respect, to arbitrary  $h \in G$ . It is well known that all finite groups<sup>1</sup> and all compact Lie groups are unimodular, as are the motion groups<sup>2</sup> considered in this paper.

In the case of the Fourier transform for the real line, the function  $u(x, p) = e^{ipx}$  plays a very important role. The Fourier transform defined according to the equation

$$\hat{f}(p) = \int_{-\infty}^{\infty} f(x) u(-x, p) dx$$

reduces the convolution equation

$$f(x) = \int_{-\infty}^{\infty} f_1(y) f_2(x - y) dy$$

to a simple product in Fourier space:

$$\hat{f}(p) = \hat{f}_1(p) \hat{f}_2(p).$$

This convolution property is based on the homomorphism property

$$u(x_1, p) u(x_2, p) = e^{ipx_1} e^{ipx_2} = e^{ip(x_1+x_2)} = u(x_1 + x_2, p).$$

The exponentials  $u(x, p) = e^{ipx}$  are, therefore, *representations* of the translation group (see [18, 33] for definitions), labeled by the continuous parameter  $p$ . They are unitary in the sense that  $u(x, p) \overline{u(x, p)} = 1$ .

An analogous function is needed for a unimodular group  $G$  if the Fourier transform pair is to be defined. However, since  $G$  is usually not commutative, a scalar function  $u(g, p)$  is not sufficient to reflect its noncommutative nature. Instead of a unitary scalar function, we must look for a unitary matrix  $U(g, p)$  with elements  $u_{mn}(g, p)$ . The variable  $p$  is called the *dual* variable and may be a scalar or other quantity. These matrices (called irreducible unitary representations of the group) must have the homomorphism property

$$U(g_1 \circ g_2, p) = U(g_1, p) U(g_2, p)$$

for all  $g_1, g_2 \in G$  and  $p$  labels inequivalent IURs of  $G$ . In addition to the homomorphism property which is required in the definition of a representation, these matrices are unitary in the usual sense ( $U^\dagger(g) U(g) = 1$ ), and irreducible in the sense that they cannot be block-diagonalized by a similarity transformation. A number of works including [33] have shown that the elements of the unitary matrices  $U(g, p)$  of many groups can be defined in terms of the standard functions of mathematical physics.

The Fourier transform of a suitable function  $f(g)$  and the inverse transform are defined as

$$\mathcal{F}(f) = \hat{f}(p) = \int_G f(g) U(g^{-1}, p) d\mu(g)$$

<sup>1</sup> In the special case of finite groups, integration is viewed as summation.

<sup>2</sup> The motion group of 3-space, SE(3), consists of all elements of the form  $g = (\mathbf{a}, A) \in \mathbb{R}^3 \times \text{SO}(3)$  with the product  $(\mathbf{a}_1, A_1) \circ (\mathbf{a}_2, A_2) = (\mathbf{a}_1 + A_1 \mathbf{a}_2, A_1 A_2)$ . The discrete motion groups are defined in an analogous way with any discrete subgroup of SO(3) playing the role of SO(3) in the definition of SE(3).

and

$$f(g) = \mathcal{F}^{-1}(\hat{f}(p)) = \int_{\hat{G}} \text{trace}(\hat{f}(p)U(g, p)) d\nu(p),$$

where  $d\nu(p)$  is an appropriately defined measure on the dual space,  $\hat{G}$ , of  $G$ .

The irreducible representations of the 3D motion group may be built on the space of square-integrable functions on the 2-sphere,  $L^2(S^2)$ , with the inner product defined as

$$(\varphi_1, \varphi_2) = \int_{\theta=0}^{\pi} \int_{\phi=0}^{2\pi} \overline{\varphi_1(\mathbf{u})} \varphi_2(\mathbf{u}) \sin \theta d\theta d\phi, \quad (2)$$

for  $\varphi_i(\mathbf{u}) \in L^2(S^2)$  where  $\mathbf{u} = (\sin \theta \cos \phi, \sin \theta \sin \phi, \cos \theta)$ , and  $0 \leq \theta \leq \pi$ ,  $0 \leq \phi \leq 2\pi$ .

The rotation subgroup  $\text{SO}(3)$  of the motion group acts on  $\hat{\mathbf{T}}$  (where  $\hat{\mathbf{T}}$  is the dual space of the translation subgroup  $\mathbf{T} = (\mathbb{R}^3, +)$ ) by rotations, so  $\hat{\mathbf{T}}$  is divided into orbits  $S_p^2$ , where  $S_p^2$  are spheres of radius  $p = |\mathbf{p}|$ .

To construct the representations of the motion group explicitly we choose a particular vector  $\hat{\mathbf{u}} = (0, 0, 1)$  on  $S^2$ . The vector  $\hat{\mathbf{u}}$  is invariant with respect to rotations from the  $\text{SO}(2)$  subgroup of  $\text{SO}(3)$

$$\Lambda \hat{\mathbf{u}} = \hat{\mathbf{u}}; \quad \Lambda \in H_{\hat{\mathbf{u}}} = \text{SO}(2), \quad (3)$$

where  $H_{\hat{\mathbf{u}}}$  is the so-called little group of  $\hat{\mathbf{u}}$ . For each  $\mathbf{u} \in S^2$  we may find  $R_{\mathbf{u}} \in \text{SO}(3)/\text{SO}(2)$ , such that

$$R_{\mathbf{u}} \hat{\mathbf{u}} = \mathbf{u}.$$

Then for any  $A \in \text{SO}(3)$  one may check that

$$(R_{\mathbf{u}}^{-1} A R_{A^{-1}\mathbf{u}}) \hat{\mathbf{u}} = \hat{\mathbf{u}}.$$

Therefore,

$$Q(\mathbf{u}, A) \triangleq (R_{\mathbf{u}}^{-1} A R_{A^{-1}\mathbf{u}}) \in H_{\hat{\mathbf{u}}}.$$

The representations of  $H_{\hat{\mathbf{u}}}$  may be taken to be of the form

$$\Delta_s: \phi \rightarrow e^{is\phi}; \quad 0 \leq \phi \leq 2\pi,$$

and  $s = 0, \pm 1, \pm 2, \dots$ .

The representations of  $\text{SE}(3)$  (which are examples of induced representations [8]) may be built using the representations of  $H_{\hat{\mathbf{u}}}$  described above.

**DEFINITION.** The unitary representations  $U^s(\mathbf{a}, A)$  of  $\text{SE}(3)$ , which act on the space of functions  $L^2(S^2)$  with the inner product (2), are defined as

$$(U^s(\mathbf{a}, A; p)\varphi)(\mathbf{u}) = e^{-ip\mathbf{u} \cdot \mathbf{a}} \Delta_s(Q(\mathbf{u}, A)) \varphi(A^{-1}\mathbf{u}), \quad (4)$$

where  $A \in \text{SO}(3)$ ,  $\Delta_s$  are representations of  $H_{\hat{\mathbf{u}}}$  and  $s = 0, \pm 1, \pm 2, \dots$ . We denote  $\mathbf{p} = p\mathbf{u}$ , where  $\mathbf{u} \cdot \mathbf{u} = 1$ .

Each representation, characterized by  $p = |\mathbf{p}|$  and  $s$ , is irreducible (they, however, become reducible if we restrict SE(3) to SO(3), i.e., when  $a = |\mathbf{a}| = 0$ ). They are unitary, because  $(U^s(\mathbf{a}, A)\varphi_1, U^s(\mathbf{a}, A)\varphi_2) = (\varphi_1, \varphi_2)$ .

The representations (4) allow us to write the matrix elements of the IURs of SE(3) in integral form and to apply FFT methods for fast numerical computations.

### 3. ALGORITHMS FOR MOTION GROUP CONVOLUTIONS USING FFTs

Here we describe an algorithm for computing 3D continuous motion group convolutions using FFTs. We use irreducible unitary representations of SE(3) (as described in the section above) to calculate Fourier matrix elements in integral form.

To write the Fourier transform in matrix form we calculate the matrix elements of  $U(g, p)$  as

$$U_{l',m';l,m}^s(\mathbf{a}, A; p) = \int_{S^2} \overline{h_{l'm'}^s(\mathbf{u})} (U(g, p) h_{lm}^s)(\mathbf{u}) d\mathbf{u},$$

where  $d\mathbf{u} = \sin\theta d\theta d\phi$  and  $h_{l'm'}^s(\mathbf{u}) = h_{l'm'}^s(\theta, \phi)$  are generalized spherical harmonics defined below in Eq. (5).

#### 3.1. Direct Fourier Transform

We use basis eigenfunctions  $h_{lm}^s(\mathbf{u})$  to write Fourier matrix elements in the integral form

$$\begin{aligned} \hat{f}_{l',m';l,m}^s(p) &= \int_{\mathbf{u} \in S^2} \int_{\mathbf{r} \in \mathbb{R}^3} \int_{R \in SO(3)} f(\mathbf{r}, R) h_{lm}^s(\mathbf{u}) \\ &\quad \times e^{ip\mathbf{u} \cdot \mathbf{r}} \overline{\Delta_s(Q(\mathbf{u}, R)) h_{l'm'}^s(R^{-1}\mathbf{u})} d\mathbf{u} d^3r dR, \end{aligned}$$

where  $dR$  is the normalized invariant integration measure on  $SO(3)$  and  $d^3r = dr_1 dr_2 dr_3$ .

The basis functions may be expressed in the form [24]

$$h_{lm}^s(\theta, \phi) = Q_{s,m}^l(\cos\theta) e^{i(m+s)\phi}, \quad (5)$$

where

$$Q_{-s,m}^l(\cos\theta) = (-1)^{l-s} \sqrt{\frac{2l+1}{4\pi}} P_{sm}^l(\cos\theta),$$

and generalized Legendre functions  $P_{sm}^l(\cos\theta)$  are given as in Vilenkin and Klimyk [34].

Under the rotation  $R$  these functions are transformed as

$$(U^s(\mathbf{0}, R; p) h_{lm}^s)(\mathbf{u}) = \Delta_s(Q(\mathbf{u}, R)) h_{lm}^s(R^{-1}\mathbf{u}) = \sum_{n=-l}^l U_{nm}^l(R) h_{ln}^s(\mathbf{u}).$$

$U_{nm}^l(R)$  are matrix elements of  $SO(3)$  representations

$$U_{mn}^l(A) = e^{-im\alpha} (-1)^{n-m} P_{mn}^l(\cos\beta) e^{-in\gamma}, \quad (6)$$

where  $\alpha, \beta, \gamma$  are  $z - x - z$  Euler angles of the rotation and  $P_{mn}^l(\cos\beta)$  is a generalization of the associated Legendre functions [33].

Thus, the Fourier transform matrix elements may be written in the form

$$\begin{aligned}\hat{f}_{l',m';l,m}^s(p) &= \int_{\mathbf{u} \in S^2} \int_{\mathbf{r} \in \mathbb{R}^3} \int_{R \in \text{SO}(3)} f(\mathbf{r}, R) h_{lm}^s(\mathbf{u}) \\ &\quad \times e^{ip\mathbf{u} \cdot \mathbf{r}} \sum_{n=-l'}^{l'} \overline{U_{nm'}^{l'}(R) h_{l'n}^s(\mathbf{u})} d\mathbf{u} d^3r dR.\end{aligned}$$

For estimates of complexity of numerical algorithms we introduce the following notations:

- $N_r$  – number of samples on  $\mathbb{R}^3$ ,
- $N_R$  – number of samples on  $\text{SO}(3)$ ,
- $N_p$  – number of samples of the  $p$  interval,
- $N_u$  – number of samples on  $S^2$ ,
- $N_F$  – total number of harmonics.

We assume that only a finite number of harmonics are required for an accurate approximation of the function  $f(g)$ . Hence only matrix elements in the range  $|s| < S$  and  $l, l' < L$  are computed. We assume also that  $L = O(S)$ . We have made the following assumptions about the number of samples in terms of  $S$ :

- $N_r = O(S^3)$ ,
- $N_R = O(S^3)$ ,
- $N_p = O(S)$ ,
- $N_u = O(S^2)$ ,
- $N_F = O(S^5)$ .

From these definitions,  $N = N_r \cdot N_R = O(S^6)$  and  $N_p \cdot N_F = O(N)$ .

Our algorithm for the numerical computation of the direct Fourier transform is as follows:

- (a) First, assuming  $f(\mathbf{r}, R)$  is compactly supported, we compute

$$f_1(R, \mathbf{p}) = \int_{\mathbb{R}^3} f(\mathbf{r}, R) e^{i\mathbf{p} \cdot \mathbf{r}} d^3r$$

using FFTs for a Cartesian lattice in  $\mathbb{R}^3$ . This integral may be computed in  $O(N_r \log(N_r))$  computations. The resulting Fourier transform is computed on a rectangular grid. We need to perform interpolation to the spherical coordinate grid in order to compute values

$$f_1(R; p, \mathbf{u}).$$

The complexity of this interpolation is discussed in the Appendix. In general,  $O(N_r \epsilon(N_r))$  computations will be required for each different sampled rotation. For high-precision numerical approximations, 3D spline interpolation technique may be used (see, e.g., [7, 29]). For  $N_s$ -point spline interpolation for all values of rotation, the complexity of interpolation is of  $O(N_s N_r N_R)$  computations, i.e.,  $\epsilon(N_r) = N_s$ . Since spline interpolation only uses a small subset of the sample points, we assume  $N_s = O(1)$ . For exactly reversible interpolation (in exact arithmetic) Fourier interpolation can be used. In this case  $\epsilon(N_r) = O((\log N_r)^2)$ , for reasons that are explained in the Appendix.

(b) Next, we perform integration on  $\text{SO}(3)$

$$(f_2)_{nm'}^{l'}(p, \mathbf{u}) = \int_{\text{SO}(3)} f_1(R, p, \mathbf{u}) \overline{U_{nm'}^{l'}(R)} dR.$$

This is the Fourier transform on the rotation group, computed for different values of  $p$  and  $\mathbf{u}$ . A fast Fourier transform technique for  $\text{SO}(3)$  has been developed by Maslen and Rockmore which calculates the forward and inverse Fourier transform of band-limited functions on  $\text{SO}(3)$  in  $O(N_R(\log N_R)^2)$  arithmetic operations for  $N_R$  sample points [23]. This may be applied to compute  $f_2$  for all values of  $p$  and  $\mathbf{u}$  and all indices in  $O(N_p N_u N_R (\log N_R)^2)$ . We assume that  $N_p N_u \approx N_r$ ; thus the order of computations is  $O(N_r N_R (\log N_R)^2)$ .

(c) Finally, we may perform integrations on the unit sphere

$$\hat{f}_{l',m';l,m}^s(p) = \sum_{n=-l'}^{l'} \int_{S^2} (f_2)_{nm'}^{l'}(p, \mathbf{u}) h_{lm}^s(\mathbf{u}) \overline{h_{l'n}^s(\mathbf{u})} d\mathbf{u}. \quad (7)$$

Using expression (5) for the basis functions we write (7) in the form

$$\begin{aligned} \hat{f}_{l',m';l,m}^s(p) &= \sum_{n=-l'}^{l'} \int_{S^2} (f_2)_{nm'}^{l'}(p, \mathbf{u}) Q_{s,m}^l(\cos \theta) Q_{s,n}^{l'}(\cos \theta) \\ &\quad \times \exp(i(m-n)\phi) d\phi \sin \theta d\theta. \end{aligned}$$

We may perform integration with respect to  $\phi$  using the FFT on  $S^1$

$$(f_3)_{l',m',n;m}(p, \theta) = \int_0^{2\pi} [(f_2)_{nm'}^{l'}(p, \phi, \theta) \exp(-in\phi)] \exp(im\phi) d\phi.$$

Integrations may be performed in  $O(N_p N_\theta S^3 N_\phi \log N_\phi)$ , where  $N_\phi = O(S)$  is the number of samplings on the  $\phi$  interval. Thus, the order of computations is  $O(N_r N_R \log N_R)$ .

Then, we perform integration with respect to  $\theta$

$$(f_4)_{l',m';l,m,n}^s(p) = \int_0^\pi [(f_3)_{l',m',n;m}(p, \theta) Q_{s,n}^{l'}(\cos \theta)] Q_{s,m}^l(\cos \theta) \sin \theta d\theta. \quad (8)$$

Using the fact that  $Q_{s,m}^l(\cos \theta)$  is  $P_{s,m}^l(\cos \theta)$  (up to a constant coefficient), the integration (summation) for all but  $l$  fixed indices may be performed using the Driscoll and Healy technique [12, 13, 23] in  $O(S^5 N_p N_\theta (\log N_\theta)^2)$ , where  $N_\theta = O(S)$  is the number of samples on the  $\theta$  interval. Thus, the order of computations is  $O(N_R^{5/3} N_r^{2/3} (\log N_R)^2)$ . The  $\theta$  integration may be performed in  $O(S^6 N_p N_\theta) = O(N_R^2 N_r^{2/3})$  operations using plain integration.

Finally, the matrix elements of the  $\text{SE}(3)$  Fourier transform may be found by the summation

$$\hat{f}_{l',m';l,m,n}^s(p) = \sum_{n=-l'}^{l'} (f_4)_{l',m';l,m,n}^s(p),$$

which is on the order of  $O(S^6 N_p) = O(N_R^2 N_p)$ .

Thus, the total order of computations of the direct Fourier transform is  $O(N_r N_R (\log(N_r) + (\log N_R)^2 + \epsilon(N_r) + N_R^{5/3} N_r^{2/3} (\log N_R)^2))$ . Under the assumption  $N_r = O(N_R)$  and using the notation  $N_r N_R = N$  ( $N$  is the total number of samples on  $\text{SE}(3)$ ) we write the leading order terms as  $O(N^{7/6} (\log N)^2 + N \epsilon(N^{1/2}))$ . For plain  $\theta$  integration (i.e., if the fast generalized Legendre transform is not used to evaluate Eq. (8)) the estimate becomes  $O(N^{4/3} + N \epsilon(N^{1/2}))$ .

### 3.2. Inverse Fourier Transform

The inverse Fourier transform integral may be written as

$$\begin{aligned} f(\mathbf{r}, R) &= \frac{1}{2\pi^2} \int_{p=0}^{\infty} \int_{\mathbf{u} \in S^2} \sum_{s,l,m,l',m'} \hat{f}_{l',m';l,m}^s(p) \overline{h_{lm}^s(\mathbf{u})} \exp(-ip\mathbf{u} \cdot \mathbf{r}) \\ &\quad \times \sum_{n=-l'}^{l'} U_{nm'}^{l'}(R) h_{l'n}^s(\mathbf{u}) p^2 dp d\mathbf{u}, \end{aligned}$$

where

$$\sum_{s,l,m,l',m'} = \sum_{s=-\infty}^{\infty} \sum_{l=|s|}^{\infty} \sum_{m=-l}^l \sum_{l'=|s|}^{\infty} \sum_{m'=-l'}^{l'}.$$

A band-limited approximation results when the restrictions  $s \leq S$  and  $l, l' \leq L = O(S)$  are imposed.

We note that  $p^2 dp d\mathbf{u} = d^3 p$ . Our algorithm for the inverse Fourier transform is as follows.

(a) We compute first

$$(g_1)_{l',m';n}^s(p, \mathbf{u}) = \sum_{l=|s|}^L \left[ \sum_{m=-l}^l \hat{f}_{l',m';l,m}^s(p) \overline{h_{lm}^s(\mathbf{u})} \right] h_{l'n}^s(\mathbf{u}) \quad (9)$$

for fixed values of  $l', m', n, s$ . The summation may be performed in the following way. We perform first the summation in square brackets. Using the expression for basis functions (5) this summation may be written as

$$(g_{11})_{l',m';l}^s(p, \theta, \phi) = \exp(-is\phi) \sum_{m=-l}^l \hat{f}_{l',m';l,m}^s(p) Q_{s,m}^l(\cos\theta) \exp(-im\phi).$$

Replacing the summation limits  $|m| \leq l$  by  $|m| \leq L = O(S)$  (and assuming that the corresponding elements  $\hat{f}_{l',m';l,m}^s(p)$  are zero for  $|m| > 1$  for given  $l$ ) we may compute this sum using the one-dimensional FFT for fixed values of other indices. This may be computed in  $O(S^4 N_p N_\theta S \log S) = O(N_R^{5/3} N_r^{2/3} \log N_R)$ . We note that the term  $\exp(-is\phi)$  is canceled with the corresponding term from  $h_{l'n}^s(\mathbf{u})$  in (9).

Then, we compute the summation

$$(g_{12})_{l',m'}^s(p, \theta, \phi) = \sum_{l=|s|}^L (g_{11})_{l',m';l}^s(p, \theta, \phi),$$

which may be performed in  $O(N_p N_\theta N_\phi S^4) = O(N_r N_R^{4/3})$  computations. The product

$$(g_1)_{l',m';n}^s(p, \mathbf{u}) = (g_{12})_{l',m'}^s(p, \theta, \phi) Q_{s,n}^{l'}(\cos \theta) \exp(in\phi) \quad (10)$$

may be computed in the same amount of computations.

Thus, the sum in Eq. (9) may be performed in  $O(N^{7/6} \log N)$  computations. Then, we interpolate from a spherical coordinate grid to a rectangular grid. This requires  $O(N_r N_R^{4/3} \epsilon(N_r)) = O(N^{7/6} \epsilon(N^{1/2}))$  operations.

Application of the Driscoll–Healy fast polynomial transform technique gives an additional saving in computation of sum (9). We define formally the  $\hat{f}_{l',m';l,m}^s(p)$  matrix elements to be zero for  $|m|, |s| > l$  and extend the limits of summation in (9) from  $l = 0$ , and from  $m = -L$  to  $m = L$ . Then the summation with respect to  $l$  may be performed first as

$$(g_{11})_{l',m';m}^s(p, \theta) = \sum_{l=0}^L \hat{f}_{l',m';l,m}^s(p) Q_{s,m}^l(\cos \theta)$$

using the fast transform technique in  $O(S^4 N_p S (\log S)^2) = O(N_R^{4/3} N_r^{2/3} (\log N_R)^2)$  computations. Then, the summation

$$(g_{12})_{l',m'}^s(p, \theta, \phi) = \sum_{m=-L}^L (g_{11})_{l',m';m}^s(p, \theta) \exp(-im\phi)$$

may be performed using the one-dimensional FFT in  $O(S^3 N_p N_\theta S \log S) = O(N_R^{4/3} N_r^{2/3} \log N_R)$ .

(b) Next we compute integrals of the form

$$(g_2)_{l',m';n}^s(\mathbf{r}) = \int_{\mathbb{R}^3} (g_1)_{l',m';n}^s(\mathbf{p}) \exp(-i\mathbf{p} \cdot \mathbf{r}) d^3 p$$

using 3D FFTs. This requires  $O(S^4 N_r \log(N_r)) = O(N^{7/6} \log N)$  operations to compute.

(c) The function  $f(\mathbf{r}, R)$  may be recovered by the summation

$$f(\mathbf{r}, R) = \sum_{s=-S}^S \left[ \sum_{l'=|s|}^L \sum_{m'=-l'}^{l'} \sum_{n=-l'}^{l'} U_{nm'}^{l'}(R) (g_2)_{l',m';n}^s(\mathbf{r}) \right].$$

The expression in square brackets is a set of inverse Fourier transforms on  $\text{SO}(3)$  for fixed values of  $\mathbf{r}$  and  $s$ . It takes  $O(N_r S N_R (\log N_R)^2) = O(N^{7/6} (\log N)^2)$  operations to compute.

Thus, the total order of computation for the inverse Fourier transform is  $O(N^{7/6} ((\log N)^2 + N \epsilon(N^{1/2})))$ .

*Convolution of functions.* The convolution integral

$$(f_1 * f_2)(g) = \int_{\text{SE}(3)} f_1(h) f_2(h^{-1} \circ g) d\mu(h) \quad (11)$$

may be written as a matrix product in Fourier space

$$(\mathcal{F}(f_1 * f_2))_{l',m';l,m}^s(p) = \sum_{j=|s|}^{\infty} \sum_{k=-j}^j (\hat{f}_2)_{l',m';j,k}^s(p) (\hat{f}_1)_{j,k;l,m}^s(p).$$

When  $f_1(g)$  and  $f_2(g)$  are band-limited in the sense defined earlier, the matrix product may be computed directly in  $O(N_p S^7) = O(N_r^{1/3} N_R^{7/3}) = O(N^{4/3})$  operations, which may be the largest time-consuming computation. We note that a fast matrix multiplication algorithm may be applied for  $n \times n = 2^m \times 2^m$  matrices, which is on the order of  $n^{\log_2 7}$  instead of  $n^3$  [27, 28, 30, 35]. Using this algorithm the matrix product may be computed in  $O(N^{(\log_2 7+1)/3})$  computations. Since fast matrix multiplications is an active research field in its own right, we characterize the order of computations for the convolution product as  $O(N^{(\gamma+1)/3})$ , where  $2 \leq \gamma \leq 3$  indicates the cost of matrix multiplication.

Therefore, the total order of computations of convolution is, at most,  $O(N^{(\gamma+1)/3}) + O(N^{7/6}(\log N)^2) + O(N^{7/6}\epsilon(N^{1/2}))$ .

Thus, when  $\epsilon(N_r) \leq O((\log N_r)^2)$ , the algorithm described above provides very considerable saving compare to the direct integration in (11), which is on the order of  $O(N_r^2 N_R^2) = O(N^2)$ .

### 3.3. A Fast Algorithm for Computing the Fourier Transform for the 2D Motion Group

Numerical algorithms for Fourier transforms on the 2D motion group were developed in [21] (continuous motion group) and [19] (discrete motion group). For completeness we describe below a new algorithm for the continuous motion-group Fourier transform using FFT methods. This algorithm is similar to the discrete motion-group algorithm used in [19] but uses  $\exp(im\phi)$  as basis functions instead of pulse functions. We begin with the representation operators for the 2D motion group

$$\mathcal{U}(g, p)\tilde{\varphi}(\mathbf{x}) = e^{-ip(\mathbf{r} \cdot \mathbf{x})}\tilde{\varphi}(A^T \mathbf{x}), \quad (12)$$

which are defined for each  $g = (\mathbf{r}, A(\theta)) \in \text{SE}(2)$ . Here  $p \in \mathbb{R}^+$ , and  $\mathbf{x} \cdot \mathbf{y} = x_1 y_1 + x_2 y_2$ . The vector  $\mathbf{x}$  is a unit vector ( $\mathbf{x} \cdot \mathbf{x} = 1$ ), so  $\tilde{\varphi}(\mathbf{x}) = \tilde{\varphi}(\cos \psi, \sin \psi) \equiv \varphi(\psi)$  is a function on the unit circle. Henceforth we will not distinguish between  $\tilde{\varphi}$  and  $\varphi$ .

Any function  $\varphi(\psi) \in L^2(S^1)$  can be expressed as a weighted sum of orthonormal basis functions as  $\varphi(\psi) = \sum_n a_n e^{in\psi}$ . Likewise, the matrix elements of the operator  $\mathcal{U}(g, p)$  are expressed in this basis as [33]

$$u_{mn}(g, p) = (e^{im\psi}, \mathcal{U}(g, p)e^{in\psi}) = \frac{1}{2\pi} \int_0^{2\pi} e^{-im\psi} e^{-i(r_1 p \cos \psi + r_2 p \sin \psi)} e^{in(\psi - \theta)} d\psi \quad (13)$$

$\forall m, n \in \mathbb{Z}$ . The inner product  $(\cdot, \cdot)$  is defined as

$$(\varphi_1, \varphi_2) = \frac{1}{2\pi} \int_0^{2\pi} \overline{\varphi_1(\psi)} \varphi_2(\psi) d\psi.$$

It is easy to see that  $(\mathcal{U}(g, p)\varphi_1, \mathcal{U}(g, p)\varphi_2) = (\varphi_1, \varphi_2)$ , and that  $\mathcal{U}(g, p)$  is, therefore, unitary with respect to this inner product.

We may express the Fourier matrix elements as an integral

$$\hat{f}_{mn}(p) = \int_{\mathbf{r} \in \mathbb{R}^2} \int_{\theta=0}^{2\pi} \int_{\psi=0}^{2\pi} f(\mathbf{r}, \theta) e^{in\psi} e^{i(\mathbf{p} \cdot \mathbf{r})} e^{-im(\psi-\theta)} d^2 r d\theta d\psi.$$

We compute the band-limited approximation of the Fourier transform for  $|m|, |n| \leq S$  harmonics. Furthermore, we assume that it is computed at  $N_p = O(S)$  points along the  $p$ -interval, and we assume that the order of sampling points in an  $\mathbb{R}^2$  region is on the order of  $S^2$  ( $N_r = O(S^2)$ ) and the number of sampling points of orientation angle  $\theta$  is  $N_R = O(S)$ . In this way the total number of points sampled in  $\text{SE}(2)$  is  $N = O(S^3)$ , which is on the same order as the total number of sample points in the Fourier domain.

We may perform first  $\mathbb{R}^2$  integration using the usual FFT

$$f_1(\mathbf{p}, \theta) = \int_{\mathbb{R}^2} f(\mathbf{r}, \theta) e^{i(\mathbf{p} \cdot \mathbf{r})} d^2 r$$

in  $O(N_R N_r \log N_r)$  computations when  $f(\mathbf{r}, R)$  is compactly supported.

Then, we perform interpolation to the polar coordinate mesh. For  $N_s$ -point spline interpolation this may be performed in  $O(N_s N_r N_R)$  computations. If Fourier interpolation is used this becomes  $O((\log N_r)^2 N_r N_R)$  computations.

The next step is to perform integration on  $\text{SO}(2)$ :

$$f_2^{(m)}(p, \psi) = \int_{\text{SO}(2)} f_1(p, \psi, \theta) e^{im\theta} d\theta.$$

This may be computed in  $O(N_r N_R \log N_R)$  computations. Then,  $\psi$  integration

$$f_{mn}(p) = \int_0^{2\pi} [f_2^{(m)}(p, \psi) e^{-im\psi}] e^{in\psi} d\psi$$

may be computed in  $O(S N_p S \log S) = O(N_r N_R \log N_R)$  computations.

We denote the total number of samples on  $\text{SE}(2)$  as  $N = N_r N_R = O(S^3)$ . Thus, the total number of arithmetic operations used to compute the direct Fourier transform is of  $O(N \log N) + O(N \epsilon(N^{1/2}))$ . When spline interpolation is used, the first term in this complexity estimate dominates for large values of  $N$ , whereas the second term dominates when Fourier interpolation is used.

It may be shown that the inverse Fourier transform may be performed in the same amount of computations. Note that this is faster than if the  $\psi$ -integration is performed first, even though that integration results in closed-form solutions for the matrix elements  $u_{mn}(g, p)$ .

The matrix product in the convolution may be performed in  $O(N^{(\gamma+1)/3})$  computations, where  $2 \leq \gamma \leq 3$  is the cost of matrix multiplication.

#### 4. FOURIER TRANSFORM FOR THE 3D “DISCRETE” MOTION GROUPS

In this section we develop fast approximate algorithms for computing the Fourier transform of functions on the discrete motion groups,  $G_{N_R}$ , which are the semi-direct product of the continuous translation group  $\mathbf{T} = (\mathbb{R}^3, +)$  and a finite subgroup  $I_{N_R} \subset \text{SO}(3)$  with  $N_R$  elements.  $I_{N_R}$  can be the icosahedral, the cubo-octahedral, or tetrahedral

rotational symmetry group. Our formulation is completely general, although in discussions of numerical implementations we focus on the icosahedral discrete motion group, where the number of elements is  $N_R = 60$ . In Subsection 4.1 the mathematical formulation is presented and in Subsection 4.2 the computational complexity of implementing Fourier transforms and convolution of functions on  $G_{N_R}$  is discussed.

#### 4.1. Mathematical Formulation

Instead of using spherical harmonics (5) as basis functions as was done when calculating matrix elements of the IURs of the continuous motion group, we now choose pulse functions  $\varphi_{N_R,n}(\mathbf{u})$  on  $S^2$ ; i.e., we subdivide the sphere into spherical regions and choose the  $\varphi$ -functions to satisfy the orthonormality relations

$$\int_{S^2} \varphi_{N_R,n}(\mathbf{u}) \varphi_{N_R,m}(\mathbf{u}) d\mathbf{u} = \delta_{nm},$$

where  $d\mathbf{u} = \sin\theta d\theta d\phi$ , and  $(\theta, \phi)$  are spherical coordinates.<sup>3</sup> For example, in the case of  $I_{60}$  we may subdivide the sphere into 20 equilateral triangles or 12 regular pentagons. These figures can be used as the support for pulse functions, but as we will see shortly, it is convenient to subdivide these regular figures so that 60 congruent (but irregular) regions,  $F_n$ , result. We then choose the orthonormal functions as

$$\varphi_{N_R,n}(\mathbf{u}) = \begin{cases} \left(\frac{N_R}{4\pi}\right)^{1/2} & \text{if } \mathbf{u} \in F_n \\ 0 & \text{otherwise.} \end{cases}$$

Here  $n = 1, \dots, N_R$  enumerates the different congruent polygonal regions on the sphere. We denote these  $\delta$ -like functions as  $\varphi_{N_R,n}(\mathbf{u}) = (4\pi/N_R)^{1/2} \delta_{N_R}(\mathbf{u}, \mathbf{u}_n)$ , where  $\mathbf{u}_n$  is a vector from the center of sphere to a point in  $F_n$ .

The matrix elements may be found in this basis as

$$U_{mn}^s(A, \mathbf{r}; p) = \int_{S^2} \varphi_{N_R,m}(\mathbf{u}) e^{ip\mathbf{u}\cdot\mathbf{r}} \Delta_s(R_{\mathbf{u}}^{-1} A R_{A^{-1}\mathbf{u}}) \varphi_{N_R,n}(A^{-1}\mathbf{u}) d\mathbf{u}. \quad (14)$$

Using the  $\delta$  function notations this integral may be written as

$$U_{mn}^s(A, \mathbf{r}; p) = \frac{4\pi}{N_R} \int_{S^2} \delta_{N_R}(\mathbf{u}, \mathbf{u}_m) e^{ip\mathbf{u}\cdot\mathbf{r}} \Delta_s(R_{\mathbf{u}_m}^{-1} A R_{A^{-1}\mathbf{u}_m}) \delta_{N_R}(A^{-1}\mathbf{u}, \mathbf{u}_n) d\mathbf{u}.$$

This integral may be approximated as

$$U_{mn}^s(A, \mathbf{r}; p) \approx 4\pi/N_R e^{ip\mathbf{u}_m\cdot\mathbf{r}} \Delta_s(R_{\mathbf{u}_m}^{-1} A R_{A^{-1}\mathbf{u}_m}) \delta_{N_R}(A^{-1}\mathbf{u}_m, \mathbf{u}_n). \quad (15)$$

We again approximate  $\delta$  functions as

$$4\pi/N_R \delta_{N_R}(A^{-1}\mathbf{u}_m, \mathbf{u}_n) = \delta_{A^{-1}\mathbf{u}_m, \mathbf{u}_n} \triangleq \begin{cases} 1 & \text{if } A^{-1}\mathbf{u}_m = \mathbf{u}_n \\ 0 & \text{otherwise,} \end{cases}$$

<sup>3</sup> The set of functions  $\{\varphi_{N_R,n}(\mathbf{u})\}$  is, of course, not complete in  $L^2(S^2)$  and therefore is not a basis in which to expand matrix elements of the IURs, but we take this into account shortly.

which means that we discretize the rotation group; i.e., we restrict rotations to the rotations  $A_j$  from the finite subgroup  $I_{N_R}$  of  $\text{SO}(3)$ , and  $A_j^{-1}\mathbf{u}_m = \mathbf{u}_n$ .

Thus, matrix elements of irreducible unitary representations of the “discrete” motion subgroup  $G_{N_R}$  are given as

$$U_{mn}^s(A_j, \mathbf{r}; p) = e^{ip\mathbf{u}_m \cdot \mathbf{r}} \Delta_s(R_{\mathbf{u}_m}^{-1} A_j R_{\mathbf{u}_n}) \delta_{A_j^{-1}\mathbf{u}_m, \mathbf{u}_n}. \quad (16)$$

Here  $s$  enumerates the representations of the little group, which is a finite subgroup  $C_n \subset \text{SO}(2)$  in this case, and  $s = 0, 1, \dots, n - 1$  for  $C_n$ . Although the expression (16) has been derived as an approximation of the continuous expression (14), the matrix elements (16) are exact expressions for the matrix elements of the irreducible unitary representations of  $G_{N_R}$ ; i.e., the relation  $U(g_1, p) \cdot U(g_2, p) = U(g_1 \circ g_2, p)$  holds.

We note  $R_{\mathbf{u}_m} \mathbf{u}_0 = A_m \mathbf{u}_0 = \mathbf{u}_m$ , where  $\mathbf{u}_0$  is the center of one of the spherical polygons  $F_0$  which is chosen arbitrarily.

However, due to the incompleteness of the set of functions  $\{\varphi_{N_R, n}\}$ , the Fourier transform matrix elements defined by (16) form an incomplete set of matrix elements. A complete set of elements are defined in the following way.

Let us allow the vector  $\mathbf{u}_m^w$  to point to an arbitrary position,  $w$ , inside the spherical shape which forms the support for the pulse basis function  $\varphi_{N_R, m}$ .

It is then possible to verify that the following matrix elements form a complete set of matrix elements for  $G_{N_R}$ :

$$U_{mn}^s(A_j, \mathbf{r}; p, w) = e^{ip\mathbf{u}_m^w \cdot \mathbf{r}} \Delta_s(R_{\mathbf{u}_m}^{-1} A_j R_{\mathbf{u}_n}) \delta_{A_j^{-1}\mathbf{u}_m, \mathbf{u}_n}. \quad (17)$$

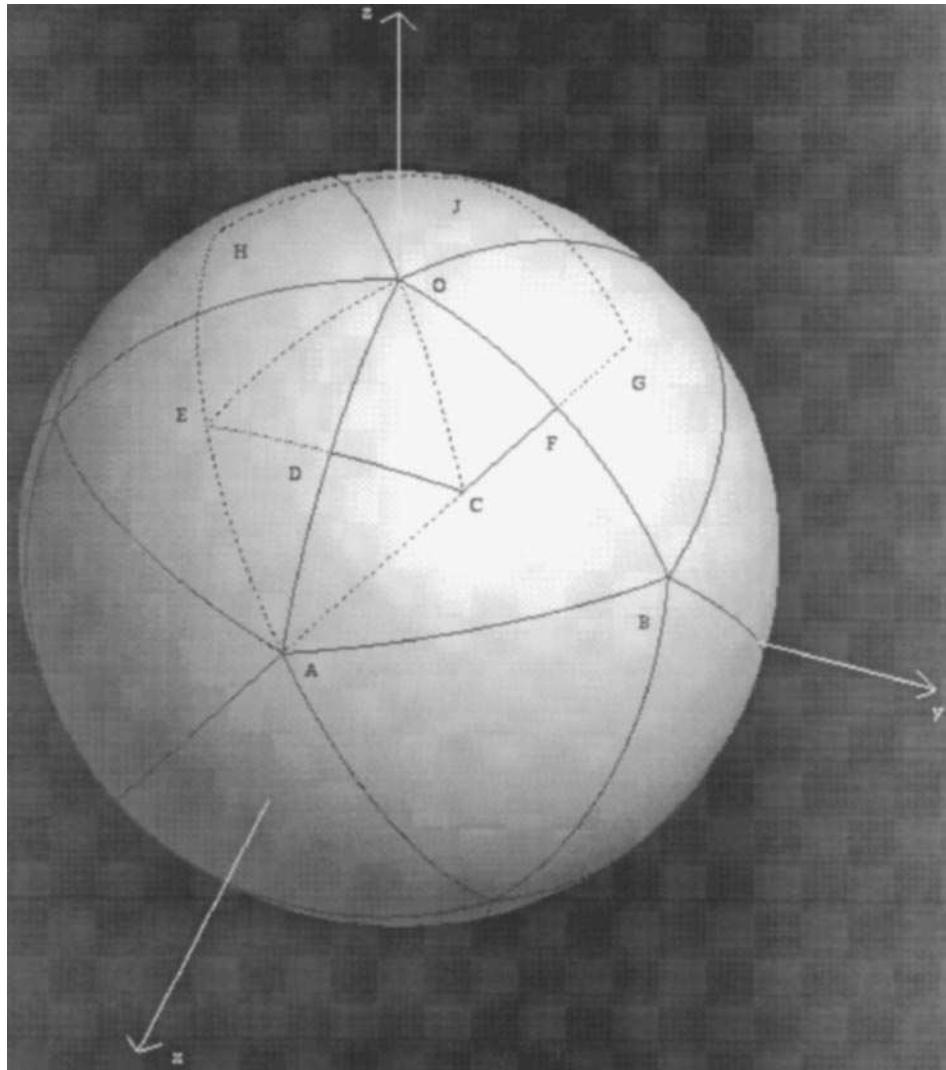
In particular, it is easy to check that  $U(g_1; p, w) \cdot U(g_2; p, w) = U(g_1 \circ g_2; p, w)$ .

As an example we consider the case when  $I_{N_R}$  is the icosahedral subgroup of the  $\text{SO}(3)$ , which has  $N_R = 60$  elements. This is the largest finite subgroup of  $\text{SO}(3)$  [18]. If we subdivide the sphere into 20 equilateral spherical triangles (see Fig. 1), this group has 6 axes of rotation of order 5 (i.e., rotation  $2\pi/5 \cdot n$ ,  $n = 0, 1, 2, 3, 4$ , around each axis) located at the triangle corners, 10 axes of order 3 located at the triangle centers, and 15 axes of order 2 located in the middle point of each triangle side.

Different representations of this discrete motion group may be classified according to different choices of little group  $C_n$ . This corresponds to choosing orthogonal pulse functions with differently shaped support. To illustrate the possible cases consider the tessellations of the sphere in Fig. 1. In order to have a complete set of matrix elements for nontrivial little group,  $C_n$ , we must consider all possible  $s = 0, \dots, n - 1$ . The representations for different  $s$  can be viewed as blocks in a  $60 \times 60$  representation matrix.

The possible choices for little groups corresponding to the shapes illustrated in Fig. 1 are:

(1) *Little group  $C_5$* : For 12 regular spherical pentagons, such as ECGJH, chosen as the support for spherical pulse functions we have  $12 \times 12$  representations. The little group of  $\mathbf{u}_0$  is  $C_5$ , thus  $s = 0, 1, 2, 3, 4$ . These representation matrices, each corresponding to an element of the little group enumerated by a value of  $s$ , may be viewed as blocks in  $60 \times 60$  representation matrices of  $G$ .



**FIG. 1.** Illustration for basis function of discrete motion group.

(2) *Little group  $C_3$* : Twenty equilateral spherical triangles are chosen as pulse functions on  $S^2$ , such as triangle ABO. We have 20 vectors  $\mathbf{u}_m$  pointing to the centers of triangles; i.e., the representations and Fourier matrices consist of  $20 \times 20$  nonzero blocks (for each fixed  $p$  and  $s$ ). The little group of the arbitrary chosen vector  $\mathbf{u}_0$  is  $C_3$ ; thus we have three inequivalent representation of the little group for each value of  $p$  enumerated by  $s = 0, 1, 2$ . These representation matrices may be combined as blocks to form  $60 \times 60$  representation matrices of  $G$ .

(3) *Little group  $C_2$* : Thirty spherical parallelograms (such as OEAC) are chosen as the support of pulse functions, and we have  $30 \times 30$  representations matrices. The little group is  $C_2$ , and  $s = 0, 1$ .

(4) *Trivial little group:* The 4-sided figure ODCF of Fig. 1 may be chosen as the support for pulse functions. We have 60 such figures; the little group is trivial in this case since they possess no rotational symmetry. The representation matrices are then  $60 \times 60$ . We note that some other divisions of the sphere into 60 equal spherical figures lead to equivalent representations (for example, the choice of the triangles ACO or ECO). These representations are irreducible.

We note that each row and column of the  $60 \times 60$  representation matrices corresponding to the trivial little group contain only one nonzero element. The matrix elements of the representations are written in this case as

$$U_{mn}(A_j, \mathbf{r}; p, w) = e^{ip\mathbf{u}_m^w \cdot \mathbf{r}} \delta_{A_j^{-1}\mathbf{u}_m, \mathbf{u}_n}. \quad (18)$$

Henceforth we restrict the discussion to this case.

The direct Fourier transform is defined as

$$\hat{f}_{mn}(p, w) = \sum_{i=0}^{N_R-1} \int_{\mathbb{R}^3} f(A_i, \mathbf{r}) U_{mn}^{-1}(A_i, \mathbf{r}; p, w) d^3 r, \quad (19)$$

where it depends now on  $w$ . The vector  $\mathbf{u}_m^w$ , which is inside the figure  $F_m$ , may be found by the rotation  $A_m$  (which transforms  $F_0$  to  $F_m$ ) from  $\mathbf{u}_0^w$ , as  $\mathbf{u}_m^w = A_m \mathbf{u}_0^w$ . The parameter  $w$ , thus, denotes the position inside the figure  $F_0$ .

The inverse Fourier transform is

$$\mathcal{F}^{-1}(\hat{f}) = \frac{1}{8\pi^3} \sum_{m=0}^{N_R-1} \sum_{n=0}^{N_R-1} \int_0^\infty \int_{F_0} \hat{f}_{mn}(p, w) U_{nm}(A_i, \mathbf{r}; p, w) p^2 dp d^2 w \quad (20)$$

(integration with respect to  $w$  is over the area of the basis figure  $F_0$ ). The vector  $\mathbf{u}_m^w$  is found by rotation from  $\mathbf{u}_0^w$ .

We choose the  $z$  axis passing through the vertex O. The  $\phi = 0$  circular arc then contains side OD and the  $\phi = 2\pi/5$  arc contains side OF. With this choice, the border of the area  $F_0$  for the 4-sided figure ODCF of Fig. 1 may be parameterized in terms of the spherical angles  $\theta, \phi$  as

$$\theta(\phi) = \arcsin \left[ \frac{0.525732}{\sqrt{1 - 0.723605 \sin^2(\phi')}} \right],$$

where

$$\phi' = \phi, \quad \text{if } 0 \leq \phi \leq \pi/5,$$

and

$$\phi' = 2\pi/5 - \phi, \quad \text{for } \pi/5 \leq \phi \leq 2\pi/5.$$

This dependence is derived using the relationships between the angles of spherical triangles (e.g., the “sin” and “cos” theorems from spherical trigonometry).

Due to the completeness of matrix elements the application of direct and inverse Fourier transform reproduces functions on the discrete motion group  $G$

$$\mathcal{F}^{-1}\mathcal{F}(f(g)) = f(g).$$

The completeness of the set of IURs we have developed for the 3D discrete motion group may be seen by first observing the integral representation for the  $\delta$  function in  $\mathbb{R}^3$ :

$$\int_{\mathbb{R}^3} e^{i\mathbf{p}\cdot\mathbf{r}} d^3 p = (2\pi)^3 \delta(\mathbf{r}).$$

Here integration is through the Fourier space, which is parameterized by  $\mathbf{p}$ .

The completeness relation

$$\begin{aligned} & \sum_{m=0}^{N_R-1} \sum_{n=0}^{N_R-1} \int_0^\infty \int_{F_0} \overline{U_{mn}(A_i, \mathbf{r}_1; p, w)} U_{mn}(A_j, \mathbf{r}_2; p, w) p^2 dp dw \\ & = (2\pi)^3 \delta^3(\mathbf{r}_1 - \mathbf{r}_2) \delta_{A_i, A_j} \end{aligned} \quad (21)$$

( $d^2w = \sin\theta d\theta d\phi$ ) then follows, because the integration is over the whole space of  $\mathbf{p} = p\mathbf{u}$  values. Repeated integration along the boundary of the basis figures  $F_i$  gives zero contribution, because the functions are nonsingular and the integration measure along the boundaries is zero.

The orthogonality relation is written as

$$\begin{aligned} & \sum_{i=0}^{N_R-1} \int_{\mathbb{R}^n} \overline{U_{mn}(A_i, \mathbf{r}; p, w)} U_{m'n'}(A_i, \mathbf{r}; p', w') d^3 r \\ & = (2\pi)^3 \frac{\delta(p - p')}{p^2} \delta_{m,m'} \delta_{n,n'} \delta^2(w - w'). \end{aligned} \quad (22)$$

Using the orthogonality relations one may check that the convolution properties and Plancherel (Parseval) identity are exact for square integrable functions on the discrete motion group.

#### 4.2. Computational Complexity

We now analyze the computational complexity of an algorithm for the numerical implementation of convolution of functions on the discrete motion group (with the icosahedral group as the rotation subgroup). In particular, we show that the convolution of functions  $f_1(\mathbf{r}, A_i)$  and  $f_2(\mathbf{r}, A_i)$  sampled at  $N = N_R \cdot N_r$  points ( $N_r$  is the number of samples in a region in  $\mathbb{R}^3$  and  $N_R$  is the order of a finite rotation subgroup, which is 60 for the icosahedral group) may be performed in  $O(N(\epsilon(N_r) + \log N_r)) + O(NN_R^{\gamma-2})$  operations (where again  $2 \leq \gamma \leq 3$  is the exponent for matrix multiplication) instead of the  $O(N^2)$  computations required for the direct computation of convolution by discretization of the convolution integral and evaluation at each discrete value of translation and rotation. The structure of matrix elements of (18) allows one to apply fast Fourier methods and reduce the amount of computations. However, even without the application of the FFT, the amount of computations required to compute convolutions using the group-theoretical Fourier transform method is  $O(N^2/N_R)$ , which is a savings over brute-force discretization of the convolution integral.

First, we estimate the amount of computations to perform the direct and inverse Fourier transforms of  $f(g)$ . We assume that we restrict  $p$  values to a finite interval and sample it at  $N_p$  points, and sample the  $w$  region at  $N_w$  points. We also assume that the total number of harmonics  $N_p N_w N_R^2 = N = N_R N_r$ .

Let us consider the direct Fourier transform (19). Each term  $i$  (for fixed  $A_i$ ) gives one nonzero term in each row and column of the Fourier matrix  $\hat{f}_{mn}^w(p)$  (because only one element in each row and column of  $U_{mn}^{-1}(g; p, w)$  is nonzero). For each fixed  $i$  we may compute the FFT of  $f(\mathbf{r}; A_i)$ , which may be computed in  $O(N_r \log(N_r))$  operations. The Fourier transform elements found by FFT are computed on a square grid of  $\mathbf{p}$  values. We, however, need to interpolate the Fourier elements computed on the grid to the Fourier elements computed in polar (spherical) coordinates. The radial part  $p$  is determined by the length of  $\mathbf{p}$ , the angular part by the indices  $m$  and  $w$  (the other index  $n$  is determined uniquely for given  $A_i$ ). If we interpolate the values from the square grid to the values on the polar (spherical) grid,  $O(N_r \epsilon(N_r) N_R)$  computations are required. Each term  $i$  in (19) may be computed in  $O(N_r \log(N_r))$  computations. The whole Fourier matrix is found in  $O(N_R N_r \log(N_r))$  computations.

Again, one element from each row and column is used in the computation of the inverse Fourier transform for each rotation element  $A_i$ . After inverse interpolation to Cartesian coordinates (which may be done in  $O(N_r \epsilon(N_r))$  computations), the inverse Fourier integration may be performed in  $O(N_R N_r \log(N_r))$  computation using the FFT. Thus, in  $O(N_R N_r \log(N_r))$  computations we reproduce the function for each  $A_i$ .

The matrix product of  $\hat{f}_{mn}(p, w)$  may be computed in  $O(N^\gamma)$  computations for each value of  $p$  and parameters  $w$ . This means that the convolution (which is a matrix product of Fourier matrices) may be performed in  $O(N_\gamma^3 N_p N_w) = O(N_R^{\gamma-2} N)$  computations.

Therefore, the convolution of functions on the discrete motion group may be performed in  $O(N \cdot (\log N_r + \epsilon(N_r))) + O(N N_R^{\gamma-2})$  using Fourier methods on the discrete motion group and the FFT. When spline interpolation is used, the term with  $\epsilon(N_r)$  is of subleading order, whereas this term is of leading order when Fourier interpolation is used.

## 5. CONCLUSIONS

Fast numerical algorithms for computing the convolution product of functions on motion groups were derived. These algorithms use the group-theoretic Fourier transform with the irreducible unitary representations written in operator form, and their matrix elements are calculated numerically instead of analytically. This, together with interpolation from Cartesian to spherical coordinate grids, made it possible to use well-known FFTs for compactly supported functions on  $\mathbb{R}^3$ , and more recent FFTs for the sphere and rotation group.

## APPENDIX: FOURIER AND SPLINE INTERPOLATION BETWEEN CARTESIAN AND POLAR GRIDS

In the complexity analysis presented in the body of this paper, the interpolation of function values between Cartesian and polar (or spherical) coordinate grids was described as an  $O(N_r \epsilon(N_r))$  computation, where  $N_r$  is the number of sample points in a bounded region of the plane (or in three dimensions). In this Appendix, we give explicit forms for  $\epsilon(N_r)$  for the cases of spline ( $\epsilon(N_r) = N_s = O(1)$ ) and Fourier ( $\epsilon(N_r) = (\log N_r)^2$ ) interpolation. We reason that Fourier interpolation is reversible in exact arithmetic, and we provide references that address the numerical error associated with spline interpolation.

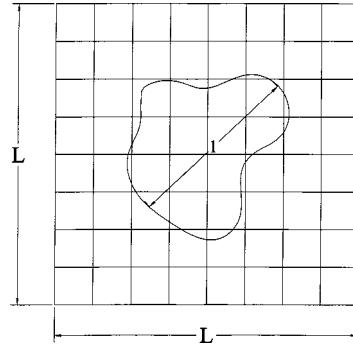
Our use of interpolation is in the frequency domain, and so the Cartesian coordinates are labeled as  $\mathbf{P} = [p_1, p_2]^T$  and polar coordinates are  $(p, \psi)$ . The discussion of interpolation between Cartesian and spherical coordinates in three dimensions follows analogously. Below we describe a Fourier interpolation method that can exactly reproduce values interpolated from Cartesian to polar to Cartesian grids, but first, the context of this result must be established.

#### A.1. The Meaning of Exactness in Fourier Analysis

To begin, let  $f(\mathbf{P})$  be a complex-valued function with compact support in  $\mathbb{R}^2$ . It is customary in numerical Fourier analysis and image processing to replace the domain of  $f$  (which is  $\mathbb{R}^2$ ) with the two-dimensional torus,  $T^2 = (\mathbb{R}/L\mathbb{R})^2$ , where  $L$  is some measure of length that is larger than the longest line segment contained in the support of  $f$  (see Fig. 2). The new function is a “folded” or “wrapped” version of  $f$ . This is equivalent to replacing  $f$  with an  $L$ -periodic version of itself defined on the original domain.

In numerical contexts, a “function” is not truly a mapping such as  $f^{(1)}: \mathbb{R}^2 \rightarrow \mathbb{C}$  or  $f^{(2)}: T^2 \rightarrow \mathbb{C}$ . Rather, it is an array of discrete values such as  $f^{(3)}: \mathbb{Z}^2 \rightarrow \mathbb{C}$ , or more precisely, a finite array of values such as  $f^{(4)}: (\mathbb{Z}/N_r^{1/2}\mathbb{Z})^2 \rightarrow \mathbb{C}$ . Often there is some confusion as to which  $f^{(k)}$  is being used. While  $f^{(1)}$  may be the function that is stated, numerical computations using the FFT are restricted to functions of the form of  $f^{(4)}$ . In order to avoid cumbersome notations, it is understood that whenever we refer to numerical computations involving  $f$ , it is always  $f^{(4)}$  (a regularly sampled array of the folded and band-limited version of  $f$ ) that is being used.

If operations such as convolutions of  $f$  with itself are to be performed numerically, then a sufficient “zero padding” must be in place around the support of  $f$ . The benefit of dealing with a zero-padded function with finite support on the torus (as opposed to the original function on the plane) is that the torus group is compact, and so a band-limited approximation to a function on the torus can be computed using the FFT. In contrast, a band-limited function on the real line (i.e., one with a spectrum that has finite support) can generally only be reconstructed from an infinite number of samples. Since the Fourier transform of such a function has finite support, then the function itself cannot have finite support.



**FIG. 2.** A function with compact support.

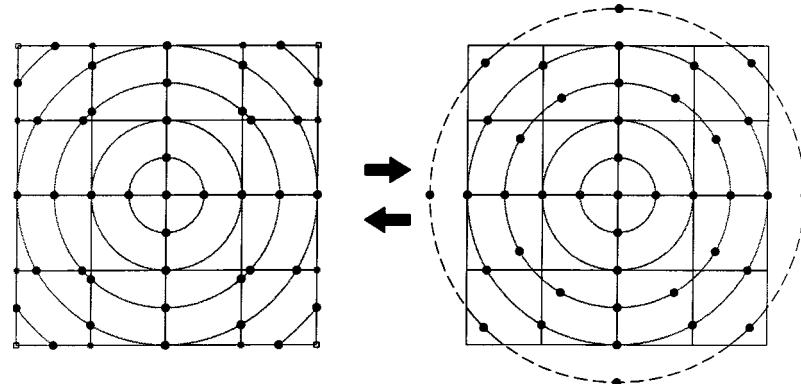
Hence, when we make the statement that the Fourier transform of a function on the plane (and the inverse Fourier transform) can be computed exactly in  $O(N_r \log N_r)$  computations, the computation is really performed on the torus, where a band-limited Fourier series (with band-limit  $N_r$ ) is sampled at  $N_r$  points on an  $N_r^{1/2} \times N_r^{1/2}$  folded version of a Cartesian grid. Both the calculations of Fourier series and Fourier coefficients are then replaced by (finite) discrete Fourier transforms.

An implicit assumption in our analysis (and every analysis we know of utilizing the classical FFT) is that the original compactly supported function  $f$  defined on the plane is sufficiently well behaved that the Fourier series of the folded version of  $f$  converges rapidly enough on the support of  $f$  (and on the surrounding zero-padded region) that for all intents and purposes the two functions are indistinguishable for the value of  $N_r$  chosen. That is, the mean-squared error between the functions is small enough to be considered negligible.

Hence, when one says that the FFT is used to compute convolutions on  $\mathbb{R}^n$  exactly at the sample points, what is meant is that the band-limited Fourier series of the folded version of the original function is a sufficiently good approximation that it takes the place of the original function. It is this approximation that often serves as the starting point of fast “exact” algorithms in computational harmonic analysis.

#### A.2. Fast Exact Fourier Interpolation

We can, without loss of any information, interpolate finite values of a periodic band-limited function on the plane (or, equivalently, a band-limited function on  $T^2$ ) sampled at Cartesian grid points to points on a polar grid. This is shown in the left-hand side of Fig. 3, where function values are specified at each intersection of the straight lines. The first step in this procedure is the interpolation of the given values to each of the points marked with circular dots. Since each straight grid line (which can be viewed as an unwrapped circle) defines a one-dimensional Fourier series, this step amounts to the evaluation of a one-dimensional Fourier series at nonequally spaced points. Since band-limited Fourier series can be viewed as polynomials of the form  $\sum_{n=0}^{N_r^{1/2}-1} a_n Z^n$ , where  $Z$  is a complex exponential, each vertical and horizontal evaluation can be performed in



**FIG. 3.** Reversible interpolation between Cartesian and polar grids.

$O(N_r^{1/2}(\log N_r^{1/2})^2)$  [1, 2]. Since there are  $O(N_r^{1/2})$  such lines, the total procedure of interpolating values from the Cartesian to polar grid requires  $O(N_r^{1/2} \cdot N_r^{1/2}(\log N_r^{1/2})^2) = O(N_r(\log N_r)^2)$  computations. This procedure is an exact one in exact arithmetic, and since we are careful to choose a polar grid for which there are a sufficient number of intersections of circles with each Cartesian grid line, no information is lost.

Once the values have been interpolated to each of the concentric circles, they define a band-limited Fourier series on each circle that can be evaluated at equally spaced angles within each circle. This is again an  $O(N_r(\log N_r)^2)$  computation since there are  $O(N_r^{1/2})$  circles. It too is exact in the sense that it conserves the original information defined at the  $N_r$ , grid points.

In order for the information of the function values to be recoverable to the Cartesian grid from the polar grid, a sufficient number of circles extending outside of the original Cartesian grid must have values defined on them. In the example shown, there is only one circle that extends beyond the Cartesian grid.

To see that this procedure is reversible, first evaluate the Fourier series on each circle at the points on the circle that intersect the Cartesian grid. Then fit the band-limited Fourier series on each straight line that passes through the values at the points of intersection. This fitting of a Fourier series to irregularly spaced data on each line is performed with the same order of computations as the evaluation of Fourier series at irregular points [1, 2]. In order for there to be enough information to reconstruct the original band-limited Fourier series on the outer edges of the Cartesian grid, the information contained in the partially outlying circle is required. For finer grids, multiple such circles are required. In the current example, the corner values could not be determined from the polar grid unless the information in the circle extending outside of the polar grid is included. The values marked with an asterisk are found using a two-step process whereby function values on surrounding lines must be evaluated at Cartesian grid points before there is enough data available to exactly recover the values.

From the above discussion, we observe that interpolation back and forth between polar and Cartesian grids can be performed with

$$\epsilon(N_r) = (\log N_r)^2.$$

This procedure of Fourier interpolation between Cartesian and polar grids is “exact” in the sense that in exact arithmetic it reversibly takes values from one kind of grid into another.

The number of function values on each circle enclosed in the Cartesian grid increases linearly with the radius. This ensures that the information in a band-limited function on  $T^2$  is exactly transferred to the collection of band-limited functions on the circles of the polar grid using the procedure described above. Another way to view this is that a band-limited periodic function on the plane is set to zero outside of a finite window with dimensions equal to the period of the function. The version of this function sampled in polar coordinates defines a new function that, in principle, can be nonzero outside of this finite window. Given the assumptions in numerical Fourier analysis stated in the previous subsection of this Appendix, the values on all circles not fully contained within the finite square grid will be made arbitrarily close to zero by appropriate choice of the band-limit and zero padding.

The circles partially enclosed in the Cartesian grid diminish in significance with radius. In the context of the example shown in Fig. 3, we see that on circles from the center outward

there are 1,4,4,12,12,8 values. This means that the band limit for functions of the angle  $\psi$  for each of these circles will be the numbers given in the previous sentence. At radius  $p$ , the band limit will be  $O(p)$ , with  $p \leq O(N_r^{1/2})$ .

### A.3. Approximate Spline Interpolation

In spline interpolation, only the  $N_s$  nearest grid points to the point at which the interpolated value is desired enter the calculation. This necessarily means that some information about the function is destroyed in the spline interpolation process. However, since the functions in question are all assumed to be band-limited Fourier series, they do not oscillate on the length scale of the distance between sample points. Hence, a polynomial spline of sufficiently high order will approximate well the local neighborhood of the point at which the interpolated value is to be determined. While this technique is an approximate one, it has been used with great success in computer tomography [7, 11, 27], image processing [10, 16, 22, 25, 32], and other fields [3, 36].

The benefit of this approach is that

$$\epsilon(N_r) = N_s = O(1).$$

The value of  $N_s$  is chosen for a given finite error threshold. The drawback is that unlike Fourier interpolation, it is not mathematically exact in exact arithmetic. For more discussion of splines see [29].

## REFERENCES

1. A. V. Aho, J. E. Hopcroft, and J. D. Ullman, "The Design and Analysis of Computer Algorithms," Addison-Wesley, Reading, MA, 1974.
2. A. Borodin and I. Munro, "The Computational Complexity of Algebraic and Numerical Problems," Elsevier, New York, MA, 1975.
3. O. M. Bucci, C. Gennarelli, and C. Savarese, Fast and accurate near-field-far-field transformation by sampling interpolation of plane polar measurements, *IEEE Trans. Antennas Propagation* **39** (1991), 48–55.
4. G. S. Chirikjian and I. Ebert-Uphoff, Numerical Convolution on the Euclidean group with applications to workspace generation, *IEEE Trans. Robot. Automat.* **14** (1998), 123–136.
5. G. S. Chirikjian, Conformational statistics of macromolecules using generalized convolution, *Comput. Theoret. Polym. Sci.*, in press.
6. G. S. Chirikjian and A. B. Kyatkin, "Engineering Applications of Noncommutative Harmonic Analysis," CRC Press, Boca Raton, FL, 2000, in press.
7. H. Choi and D. C. Munson, Direct-Fourier reconstruction in tomography and synthetic aperture radar, *Int. J. Imag. Syst. Technol.* **9** (1998), 1–13.
8. A. J. Coleman, "Induced Representations with Applications to  $S_n$  and  $GL(n)$ ," Queen's University, Kingston, Ontario, 1966.
9. J. W. Cooley and J. Tukey, An algorithm for the machine calculation of complex Fourier series, *Math. Comput.* **19** (1965), 297–301.
10. P.-E. Danielsson and M. Hammerin, High-accuracy rotation of images, *CVGIP: Graphical Models Image Process.* **54** (1992), 340–344.
11. S. R. Deans, "The Radon Transform and Some of Its Applications," Wiley, New York, 1983.
12. J. R. Driscoll and D. Healy, Computing Fourier transforms and convolutions on the 2-sphere, *Adv. Appl. Math.* **15** (1994), 202–250.

13. J. R. Driscoll, D. Healy, and D. N. Rockmore, Fast discrete polynomial transform with applications to data analysis for distance transitive graphs, *SIAM J. Comput.* **26** (1997), 1066–1099.
14. I. Ebert-Uphoff and G. S. Chirikjian, Efficient workspace generation for binary manipulators with many actuators, *J. Robot. Syst.* **12** (1995), 383–400.
15. D. F. Elliott and K. R. Rao, “Fast Transforms: Algorithms, Analyses, Applications,” Academic Press, New York, 1982.
16. D. Fraser and R. A. Schowengerdt, Avoidance of additional aliasing in multipass image rotations, *IEEE Trans. Image Process.* **3** (1994), 721–735.
17. J. P. Gauthier, G. Barnard, and M. Sibermann, Motion and pattern analysis: Harmonic analysis on motion groups and their homogeneous spaces, *IEEE Trans. Syst. Man Cybern.* **21** (1991), 159–172.
18. D. Gurarie, “Symmetry and Laplacians. Introduction to Harmonic Analysis, Group Representations and Applications,” Elsevier, Amsterdam, 1992.
19. A. B. Kyatkin and G. S. Chirikjian, Template matching as a correlation on the discrete motion group, *Comput. Vision Image Understanding* **74** (1999), 22–35.
20. A. B. Kyatkin and G. S. Chirikjian, An operational calculus for the Euclidean motion group with applications in robotics and polymer science, *J. Fourier Anal. Appl.*, in press.
21. A. B. Kyatkin and G. S. Chirikjian, Synthesis of binary manipulators using the Fourier transform on the Euclidean group, *ASME J. Mech. Design* **21** (1999), 9–14.
22. K. G. Larkin, M. A. Oldfield, and H. Klemm, Fast Fourier method for the accurate rotation of sampled images, *Opt. Commun.* **139** (1997), 99–106.
23. D. K. Maslen and D. N. Rockmore, Generalized FFTs — A survey of some recent results, *DIMACS Ser. Disc. Math. Theoret. Comput. Sci.* **28** (1997), 183–237.
24. W. Miller, “Lie Theory and Special Functions,” Academic Press, New York, 1968; and W. Miller, Some applications of the representation theory of the Euclidean group in three-space, *Commun. Pure Appl. Math.* **17** (1964), 527–540.
25. A. W. Paeth, A fast algorithm for general raster rotation, in “Graphics Gems” (A. S. Glassner, Ed.), pp. 179–195, Academic Press, Boston, 1990.
26. S. X. Pan and A. C. Kak, A computational study of reconstruction algorithms for diffraction tomography: Interpolation versus filtered back-propagation, *IEEE Trans. Acoust. Speech Signal Process.* **ASSP-31** (1983), 1262–1275.
27. V. Pan, How can we speed up matrix multiplication, *SIAM Rev.* **26** (1984), 393–416.
28. V. Pan, “How to Multiply Matrices Fast,” Springer-Verlag, Berlin, 1984.
29. H. Speath, “Two Dimensional Spline Interpolation Algorithms,” AK Peters, Wellesley, 1995.
30. V. Strassen, Gaussian elimination is not optimal, *Numer. Math.* **13** (1969), 354–356.
31. J. Talman, “Special Functions,” W. A. Benjamin, Amsterdam, 1968.
32. M. Unser, P. Thévenaz, and L. Yaroslavsky, Convolution-based interpolation for fast, high-quality rotation of images, *IEEE Trans. Image Process.* **4** (1995), 1371–1381.
33. N. J. Vilenkin, Bessel functions and representations of the group of Euclidean motions, *Uspekhi Mat. Nauk* **11** (1956), 69–112, [In Russian]
34. N. J. Vilenkin and A. U. Klimyk, “Representation of Lie Group and Special Functions,” Vols. 1–3, Kluwer Academic, The Netherlands, 1991.
35. S. Winograd, A new algorithm for inner products, *IEEE Trans. Comput.* **C-17** (1968), 693–694.
36. A. D. Yaghjian and M. B. Woodworth, Sampling in plane-polar coordinates, *IEEE Trans. Antennas Propagation* **44** (1996), 696–700.